UNIVERSITÉ DU QUÉBEC À MONTRÉAL

DAIRY PROFITABILITY PREDICTION USING DEEP LEARNING MODELS

THESIS

PRESENTED

AS PARTIAL REQUIREMENT

TO THE DOCTORATE IN COMPUTER SCIENCE

BY

VAHID NAGHASHI

MARCH 2026

UNIVERSITÉ DU QUÉBEC À MONTRÉAL

PRÉDICTION DE LA RENTABILITÉ LAITIÈRE À L'AIDE DE MODÈLES D'APPRENTISSAGE PROFOND

THÈSE

PRÉSENTÉE

COMME EXIGENCE PARTIELLE

DU DOCTORAT EN INFORMATIQUE

PAR

VAHID NAGHASHI

MARS 2026

ACKNOWLEDGEMENTS

I would like to express my warmest gratitude to my parents for their unconditional love, endless support and encouragement throughout this journey. Their belief in me and valuable advices have been a constant source of strength and motivation during challenging times.

I am also very grateful to my sister for always being by my side, offering their love, moral support, and valuable advice.

I would like to express my sincere appreciation to my supervisor Professor Abdoulaye Banire Diallo, for his valuable guidance, patience, and expertise. Your mentorship has played a pivotal role in the successful completion of this thesis.

My heartfelt thanks also go to my co-supervisor, Professor Mounir Boukadoum, for the constructive feedback, scientific guidance, and continued support offered throughout this work.

Finally, I would like to thank everyone who contributed directly or indirectly to this thesis.

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

ACRONYMES

UQAM Université du Québec à Montréal.

SNCF Société Nationale des Chemins de Fer.

iGRU inverted Gating Recurrent Unit.

LSTM Long-Short Term Memory.

SCC Somatic Cell Count.

MUN Milk Urea Nitrogen.

BHB Beta Hydroxybutrate.

DIM Days In Milk.

CCD Condition Code.

LACT.NO Lactation Number.

MLKNG.FQCY Milking Frequency.

MLKNG.PTRN Milking Pattern.

RÉSUMÉ

In precision livestock management, effective decision-making around animal replacement depends on accurately estimating the lifetime profitability of each animal. In dairy farming, milk production is influenced by various factors including milk quality, health conditions, genetics, and herd management practices, all of which may be affected by broader operational and market dynamics. This thesis formulates the estimation of milk production income as a multivariate time series forecasting problem, where future profitability, typically measured during later lactation periods (e.g. lactation 2 or 3) is predicted using data collected during early lactation period(s) (e.g., lactation 1). The available data includes a range of temporally ordered dairy-related features collected over several months after the cow's birth. This spatio-temporal dataset can be treated as a multivariate time series, where each variable (or channel) represents a distinct dairy factor evolving over time. Addressing this forecasting task requires models that can simultaneously capture temporal dependencies and inter-feature (cross-channel) correlations.

This thesis investigates deep learning architectures for time series forecasting, with a specific emphasis on their application to dairy income prediction. After reviewing and exploring major existing approaches, including autoregressive models, recurrent neural networks (RNNs), linear models, and Transformer-based models, we introduce several novel architectures tailored to multivariate forecasting. First, we propose an LSTM-derived architecture enhanced with an attention layer to model sequential and contextual patterns in the context of dairy prediction. Second, we introduce two Transformer-based models : one focusing on multi-scale temporal and cross-channel modeling, with another emphasizing cross-channel interactions and integrating them with captured temporal patterns. Finally, we present a recurrent model that applies GRUs bidirectionally along the channel dimension to capture complex inter-feature dependencies, particularly in dairy forecasting scenarios.

All proposed models are extensively evaluated on both publicly available benchmark datasets and real-world dairy farm data. Results underscore that our methods consistently outperform or remain competitive with current state-of-the-art models in terms of forecasting accuracy and computational efficiency.

Beyond methodological contributions, this work supports improved resource allocation and decision-making in the dairy industry, providing a data-driven foundation for more accurate profitability forecasting and cost-effective herd management.

Keywords : Lifetime profit, Time series forecasting, Dairy factors, Long-Short Term Memory (LSTM), Gated Recurrent Unit (GRU), Transformer, Prediction accuracy

RÉSUMÉ

Dans la gestion de précision de l'élevage, une prise de décision efficace concernant le remplacement des animaux dépend d'une estimation précise de la rentabilité à vie de chaque animal. En production laitière, la quantité de lait produite est influencée par divers facteurs, notamment la qualité du lait, l'état de santé, la génétique et les pratiques de gestion du troupeau, lesquels peuvent être affectés par des dynamiques opérationnelles et de marché plus larges. Cette thèse formule l'estimation des revenus de production laitière comme un problème de prévision de séries temporelles multivariées, où la rentabilité future — généralement mesurée lors des lactations ultérieures (par exemple, lactation 2 ou 3) — est prédite à partir des données recueillies pendant les premières périodes de lactation (par exemple, lactation 1).

Les données disponibles comprennent un ensemble de variables laitières ordonnées temporellement, collectées sur plusieurs mois après la naissance de la vache. Ce jeu de données spatio-temporel peut être traité comme une série temporelle multivariée, où chaque variable (ou canal) représente un facteur laitier distinct évoluant dans le temps. La résolution de cette tâche de prévision nécessite des modèles capables de capturer simultanément les dépendances temporelles et les corrélations inter-caractéristiques (entre canaux).

Cette thèse explore des architectures d'apprentissage profond pour la prévision de séries temporelles, en mettant particulièrement l'accent sur leur application à la prédiction des revenus laitiers. Après avoir passé en revue les approches existantes, notamment les modèles autorégressifs, les réseaux de neurones récurrents (RNN), les modèles linéaires et les modèles basés sur les Transformers, nous introduisons plusieurs architectures nouvelles conçues pour la prévision multivariée. Premièrement, nous proposons une architecture dérivée des LSTM, enrichie d'une couche d'attention pour modéliser les motifs séquentiels et contextuels dans le cadre de la prédiction laitière. Deuxièmement, nous introduisons deux modèles basés sur les Transformers : l'un axé sur la modélisation temporelle multi-échelle et inter-canaux, et l'autre mettant l'accent sur les interactions entre canaux tout en les intégrant aux motifs temporels capturés. Enfin, nous présentons un modèle récurrent qui applique des GRU de manière bidirectionnelle le long de la dimension des canaux, afin de capter les dépendances complexes entre caractéristiques, en particulier dans les scénarios de prévision laitière.

Tous les modèles proposés sont évalués de manière approfondie sur des jeux de données de référence publics ainsi que sur des données réelles provenant d'exploitations laitières. Les résultats montrent que nos méthodes surpassent ou égalent systématiquement les modèles de pointe actuels en termes de précision de prévision et d'efficacité computationnelle.

Au-delà des contributions méthodologiques, ce travail soutient une meilleure allocation des ressources et une prise de décision plus éclairée dans l'industrie laitière, en offrant une base fondée sur les données pour une estimation plus précise de la rentabilité et une gestion du troupeau plus rentable.

Mots-clés : Rentabilité à vie, Prévision de séries temporelles, Facteurs laitiers, Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Transformer, Précision de prédiction

INTRODUCTION

Accurate forecasting of dairy milk production is necessary for effective decision-making in modern livestock management. In dairy farming, predicting future milk yield plays a critical role in planning resource allocation, managing herd health, optimizing feeding strategies, and making economically sound culling or replacement decisions. Milk production is influenced by a combination of various factors, including environmental conditions, genetic traits, management practices, and health indicators, many of which evolve over time and exhibit complex interdependencies. These characteristics make dairy production an ideal candidate for time series forecasting approaches. By leveraging advanced deep learning architectures and adapting them for time series forecasting task, this research aims to improve the accuracy and robustness of milk production forecast, ultimately contributing to more sustainable and profitable dairy operations.

0.1 Time series

A time series is a sequence of data points collected over time, typically at regular intervals, such as monthly dairy income from a farm or milk production over a season. Time series data carries a chronological order, making it a powerful tool for capturing trends and seasonality, which helps to predict next events. Time series forecasting is defined as analyzing the sequence of data in order to identify trends and patterns, leading to build an efficient prediction model. Time series forecasting plays an essential role in many fields, including finance, agriculture, meteorology and energy consumption domains (Naghashi *et al.*, 2025a). Forecasting is treated as training models based on historical data and using them to predict future observations. A time series consists of four main components : 1- Trend : The long-term increase or decrease in the series, often modeled as a linear or nonlinear function of time, 2- Seasonality : The repeating patterns or cycles over time (recurring patterns or cycles within fixed periods e.g., higher milk production in certain months)., 3- Cyclicality : The long-term fluctuations that occur over extended periods, typically without a fixed or predictable frequency (e.g., economic cycles affecting dairy prices). 4- Noise : The random variability in the observations that cannot be captured by the model. The above mentioned elements can be considered to be combined in some way to provide the observed time series. For instance, they can be added together to form the time series or they can be multiplied, giving the resultant series. These assumptions are useful to model and analyze a time series observation. Time series forecasting approaches are divided into two main groups : univariate and multivariate models, which will be explained in the following subsections.

0.1.1 Time series as a regression task

Time series forecasting can be defined as a regression problem (Martínez *et al.*, 2022; Hyndman et Athanasopoulos, 2018). In a regression perspective, time series forecasting is treated as modeling the relationship between a dependent variable (target) and one or more independent variables (features). In the case of time series data, the dependent variable is typically the value of the series at a future time step, while the independent variables often include lagged values (for example, dairy income in the current month may depend on income from the previous month or the same month in the previous year), exogenous variables (external factors), engineered features based on trend and seasonality (a binary feature could indicate whether the current month is part of a high-demand season or not). The general form of a regression model for time series is described as follows :

$$y_t = f(y_{t-1}, y_{t-2}, \ldots, y_{t-p}, X_t) + \epsilon_t \tag{1}$$

Where,

- y_t : Target variable (value of the time series at time t).
- $y_{t-1}, y_{t-2}, \ldots, y_{t-p}$: Lagged values of the time series (features).
- X_t : Exogenous variables at time t.
- $f(\cdot)$: Regression function to be learned.
- ϵ_t : Error term (noise).

Different regression models can be applied for time series forecasting, including Random Forest, Support Vector Regression and Linear Regression (Zhang *et al.*, 1998). In the context of dairy forecasting, lagged values refer to last dairy income values. Dairy income is referred to the economic return from milk production, generally calculated as milk volume multiplied by market price, often adjusted for quality factors such as fat and protein content. Exogenous variables include external factors such as feed prices, weather conditions, and global market trends. In addition, engineered features are features like month-of-year or season-of-year indicators which can help model seasonality and trends. However, regression models ignore the auto-correlation property of time series, and the non-stationary (variables statistics over time) of time series data.

0.1.2 Time series as a forecasting problem

Time series forecasting involves predicting future values of a variable based on its past characteristics. There are various approaches for time series forecasting, ranging from traditional statistical methods to modern machine learning and deep learning techniques.

0.1.2.1 Traditional methods

Traditional statistical methods are widely used for time series forecasting due to their simplicity and interpretability (Hyndman et Athanasopoulos, 2018). These methods include (Makridakis *et al.*, 2018) : 1- Moving Averages : A simple technique that reduces short-term fluctuations by averaging past observations. 2- Exponential Smoothing : A weighted moving average method that assigns more importance to recent observations. 3- Decomposition : Separates a time series into its trend, seasonal, and residual components for improved and more accurate analysis. The traditional methods are effective for simple time series but may struggle with complex or nonlinear relationships present in real world datasets.

0.1.2.2 Machine learning approaches

Machine learning (ML) models can capture complex patterns in time series data (Bontempi *et al.*, 2012). Unlike traditional statistical algorithms, which often rely on linearity or pre-defined data distributions, ML models can learn non-linear and high-dimensional relationships directly from data (Zhang, 2003). For instance, linear regression represents the relationship between the target variable and lagged values or exogenous features. Tree-Based Models, such as decision trees (Quinlan, 1986), random forests (Breiman, 2001), and gradient boosting machines (Chen et Guestrin, 2016) handle nonlinear relationships and interactions between features. Support Vector Regression (SVR) are effective for capturing complex patterns but computationally expensive for large datasets (Makridakis *et al.*, 2018). ML models are particularly useful when exogenous variables (e.g., feed prices, weather conditions) are incorporated into the forecasting process.

0.1.2.3 Deep learning approaches

Deep learning models have gained popularity for time series forecasting due to their ability to model complex, nonlinear and long-term sequential dependencies. Common DL models include (Lim et Zohren, 2021) : Recurrent Neural Networks (RNNs) which are designed to handle sequential data by maintaining a hidden

state that captures temporal dependencies (Hewamalage *et al.*, 2021). Long Short-Term Memory (LSTM) Networks are a variant of RNNs that solve the vanishing gradient problem and better suited for capturing long-term dependencies. Gated Recurrent Units (GRUs) are similar to LSTMs but with a simpler architecture and fewer parameters. Transformers (Vaswani *et al.*, 2017) were originally developed for natural language processing, but have shown promise in time series forecasting due to their ability to capture long-range dependencies thanks to the self-attention mechanism. Deep learning models are specifically well-suited for dairy income forecasting, because they can deal with large datasets, capture seasonal and trend patterns, and incorporate exogenous variables.

0.1.3 Dealing with auto-regressive models

Auto-regressive (AR) models are a class of statistical models that use past observations of a time series to predict future values. These models are widely used in different time series forecasting applications due to their simplicity and effectiveness. Auto-regressive models assume that the value of a time series at a given time step depends linearly on its past values (Box *et al.*, 2015). The general form of an AR model of order p is described as :

$$y_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \cdots + \beta_p y_{t-p} + \epsilon_t \tag{2}$$

In which,

- y_t : The value of the time series at time t.
- β_0 : The intercept or constant term.
- $\beta_1, \beta_2, \ldots, \beta_p$: The coefficients for the lagged values $y_{t-1}, y_{t-2}, \ldots, y_{t-p}$.
- $y_{t-1}, y_{t-2}, \ldots, y_{t-p}$: The lagged values of the time series (past observations used as features).
- ϵ_t : The error term (noise or unexplained variability) at time t.

The Auto-Regressive Integrated Moving Average (ARIMA) model (Box *et al.*, 2015) extends the AR model by incorporating both differencing (to handle non-stationarity) and moving average terms. It is typically defined as **ARIMA(**p, d, q**)**, where :

- p : order of the autoregressive (AR) component,
- d : order of differencing (the Integrated component),
- q : order of the moving average (MA) component.

The **auto-regressive** part uses past values to predict upcoming ones :

$$\text{AR}(p): \quad y_t = \beta_1 y_{t-1} + \beta_2 y_{t-2} + \cdots + \beta_p y_{t-p} \tag{3}$$

The **Integrated (I)** component refers to differencing the time series d times to achieve stationarity. This is defined as :

$$\Delta^d y_t = (1 - L)^d y_t \tag{4}$$

where L is the lag operator. For example, for $d = 1$, differencing yields :

$$\Delta y_t = y_t - y_{t-1} \tag{5}$$

The **Moving Average (MA)** component models the influence of past forecast errors ϵ_{t-i}, weighted by parameters θ_i :

$$\text{MA}(q): \quad y_t = \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} \tag{6}$$

Here, the θ_i coefficients capture how previous errors contribute to the current value, enabling the model to correct for short-term fluctuations or noise.

SARIMA (Box *et al.*, 2015) is an extension of ARIMA that takes seasonality (repeating patterns at fixed intervals, like monthly or yearly cycles) into account. For instance, dairy income might spike every December due to holiday demand. The key features of SARIMA is capturing of the relationship between the current value and past values at the same point in previous seasons, removing seasonal trends by differencing the data at the seasonal intervals. Additionally, it uses past seasonal forecast errors to improve predictions. For example, if the model underestimated dairy income last December, the seasonal MA component deals with that error.

0.1.4 Univariate forecasting

In univariate approaches (Hyndman et Athanasopoulos, 2018) the future values of a target series are predicted using only one time series corresponding to a variable in the problem domain. In other words, the input is a single sequence of values regarding measurements of a feature or factor over time. For example, in our case, the sequence of dairy profits related to months in the first lactation defines a univariate time

series. Uni-RNN (Frasco *et al.*, 2020) and N-BEATS (Oreshkin *et al.*, 2019) models correspond to univariate time series prediction category. Both models are based on deep neural networks and take a single series associated with values of a factor along a time sequence.

0.1.5 Multivariate forecasting

Multivariate time series consists of two or more series related to multiple variables which are captured over a time sequence, such as a collection of dairy factors measured over consecutive months (Hyndman et Athanasopoulos, 2018). In other words, multivariate time series are associated with more than one time-dependent variable. Each variable (variate) not only depends on its own past values but also has some relations with the other variables. These dependencies should be considered in prediction of future values. Time series data exhibit both auto-correlations within a single series and cross-correlations among different series at various time steps (commonly known as cross-channel or cross-variate dependencies). Leveraging these dependencies allows a time series forecasting model to deliver more accurate predictions. For instance, in Vector Auto-Regression (VAR) model (Lütkepohl, 2005), each variable is a linear function of its past values and the previous values of all the other variables. Recurrent Neural Network (RNN)-based models effectively represent temporal dependencies across time steps using recurrent cells (Goodfellow *et al.*, 2016). However, they often fall short to account for spatial dependencies between variables. The dairy features corresponding to a sequence of months (sequence of dairy factors measured during the test dates) can be treated as a multivariate time series, with each series related to a sequence of a specific dairy factor. A test-date in a month consists of different features including quality of milk (fat, protein, lactose), management factors (milking frequency and pattern), health indicators (Somatic Cell Counts, Levels of Beta-HydroxyButyrate and Milk Urea Nitrogen).

0.1.6 Deep learning

Deep learning has emerged as a powerful paradigm within artificial intelligence, enabling significant advancements in various domains, including computer vision, natural language processing, and time series forecasting (Goodfellow *et al.*, 2016; LeCun *et al.*, 2015). Deep learning is invented based on artificial neural networks and deep learning models utilize multiple layers of nonlinear transformations to learn hierarchical and meaningful representations from raw data. The ability to automatically capture relevant features from large datasets has made deep learning an effective tool in different tasks where traditional machine learning approaches struggle to generalize. Architectures such as Convolutional Neural Networks (CNNs)

have revolutionized image processing (LeCun *et al.*, 1998), while Recurrent Neural Networks (RNNs), such as Long Short-Term Memory (LSTM) networks (Hochreiter et Schmidhuber, 1997a) and Gated Recurrent Units (GRUs) (Cho *et al.*, 2014), have demonstrated promising performance in sequential data processing. More recently, Transformer-based models (Vaswani *et al.*, 2017) have surpassed RNNs in many sequence modeling tasks due to their ability to capture long-range dependencies. In time series forecasting, deep learning models have shown remarkable promise by capturing intricate temporal correlations and cross-variable interactions. Unlike traditional statistical methods, which usually rely on predefined assumptions about data distributions and linear dependencies, deep learning approaches learn intricate and nonlinear relationships directly from provided data.

Deep learning still faces some challenges, including high computational resource demands, data requirements, and careful hyper-parameter tuning (Strubell *et al.*, 2019; Sun *et al.*, 2017). However, current advancements in model efficiency (Han *et al.*, 2015), transfer learning (Ruder *et al.*, 2019), and federated learning (Yang *et al.*, 2019) continue to push the boundaries of what deep learning can achieve. As a result, deep learning remains a main part of modern AI research, driving innovations in both academia and industry (LeCun *et al.*, 2015; Goodfellow *et al.*, 2016).

0.1.7 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a category of neural networks designed for sequential data processing across multiple time steps (Lipton *et al.*, 2015). The Gated Recurring Unit (Cho *et al.*, 2014), introduced in 2014, is a specific type of RNN with a gating mechanism to input or forget information along a sequence of time steps, which employs update and forget units to process sequential data mapped to a hidden space. A GRU cell is illustrated in Figure 0.1.

The GRU operation is defined by the following equations :

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{7}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{8}$$

$$\hat{h}_t = \phi(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{9}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \tag{10}$$

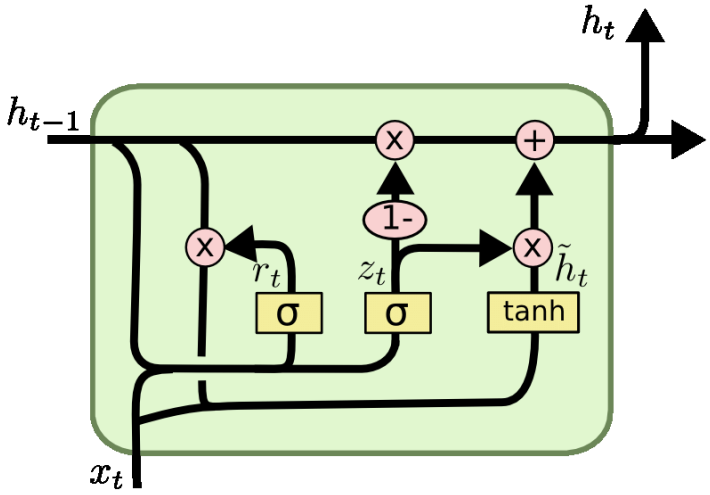


Figure 0.1 – Illustration of a Gated Recurrent Unit (GRU) cell. The GRU combines the current input x_t and the previous hidden state h_{t-1} to produce the updated hidden state h_t. It utilizes two gates : the *reset gate* r_t, which controls how much of the past information to forget, and the *update gate* z_t, which decides how much of the new candidate hidden state $\tilde{h}_t$ contributes to the final output.

Where, $\odot$ represents element-wise multiplication. x_t and h_t indicate input and hidden state vectors at time t, respectively. z_t, r_t, $\hat{h}_t$ represent the update vector, reset vector and candidate hidden state at time step t, respectively. In addition, W_h, W_z, W_r, U_h, U_z, U_r, b_h, b_z and b_r are the relevant weights and biases which are learned during the model training phase. This type of RNN showed effective sequential modeling in various applications (Wang *et al.*, 2021a), (Lawi *et al.*, 2022). In most applications, GRUs and LSTMs are commonly used to capture temporal dependencies (Lin *et al.*, 2023). However, their efficiency in modeling cross-channel correlations within time series data has often been overlooked.

0.1.8 Transformer

The Transformer model (Vaswani *et al.*, 2017) has proven its efficiency in capturing long-term dependencies in a sequence (sentence or time series), thanks to its self-attention block, which provides the correlations between different positions in a sequence. This allows to specify the importance of each input word or token to generate the output representation, and focus on the relevant elements. Moreover, a multi-head attention mechanism processes the input sequence from several viewpoints through various head dimension, enabling the model to capture different types of relationships in different projection spaces. To compute the multi-head self-attention, the input sequence is first converted to so called queries, keys, and value

matrices through linear projections :

$$Q_i = X \cdot W_i^Q, \quad K_i = X \cdot W_i^K, \quad V_i = X \cdot W_i^V \tag{11}$$

In the above formulas, X represent the embedded input sequence, Q_i, K_i, and V_i are the projection results, i.e., the query, key, and value matrices. W_i^Q, W_i^K, and W_i^V are the corresponding projection matrices for ith head. These matrices essentially project the input tokens into a shared space where they can be compared. Each projection matrix is of size $d_{model} * d_k$, where d_{model} is the embedding dimension and d_k is the head dimension. The input multivariate time series is represented in a tensor including the samples with the variables (series values) over the input time steps. Thus, X is of dimension $(Batch, Length, d_{model})$. In this context, "batch" is referred to as a collection of multiple samples, "Lengths" corresponds to the length of the input time series, and d_{model} signifies the dimension of the feature space, in which the attributes are projected.

The head dimension is calculated as $d_k = d_{model}/h$, where h is the number of heads. The projected queries and keys are multiplied, and the result is scaled down by d_k before feeding it to a softmax function in order to obtain the attention scores :

$$Attention(Q, K, V) = softmax\left(\frac{Q \cdot K^T}{\sqrt{d_k}}\right) \cdot V \tag{12}$$

Dividing the dot products of queries and keys by the square root of the head dimension ensures stability during training and controlling the scale of the attention scores.

As mentioned, the input undergoes h projections to the head space, each time with an independent self-attention computation :

$$head_i = Attention(XW_i^Q, XW_i^K, XW_i^V) \tag{13}$$

The results are concatenated to represent the temporal dependencies from different attention heads perspectives, thus providing valuable information to the downstream blocks or tasks. Finally, the concatenated

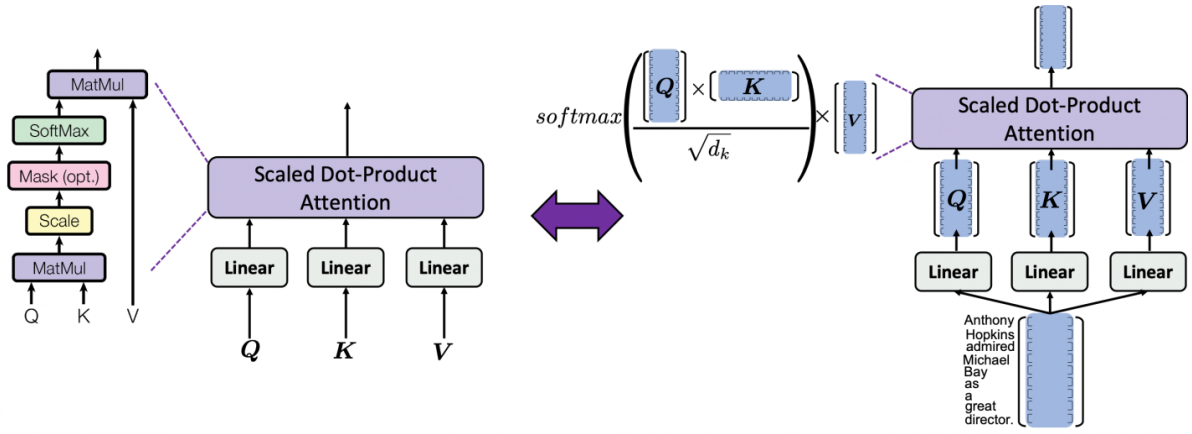


Figure 0.2 – Illustration of the multi-head self-attention mechanism

attention scores are linearly projected by a W^O matrix of size $d_{model} * d_{model}$ to convert them into an output that can be processed by additional Transformer layers :

$$MultiHead(Q, K, V) = Concat\left(head_1, \ldots, head_h\right) \cdot W^O \tag{14}$$

Figure 0.2 illustrates the main steps involved in the computation of multi-head self-attention.

0.1.9 General challenges of time series forecasting

Time series forecasting involves predicting future values based on historical data. Despite significant progress in forecasting methods, several challenges persist, making accurate predictions challenging. One of the primary challenges is the presence of non-stationarity, where the statistical properties of the time series, such as mean and variance, fluctuate over time. This prevents models to generalize well to future data or test data. Additionally, time series data often exhibit seasonality and trend components, which must be carefully represented to avoid biased predictions. Another challenge is the presence of noise or irregular fluctuations in the data, which can obscure underlying patterns and reduce the accuracy of forecasts. Real-world time series data may involve high dimensionality and intricate dependencies, particularly, when multiple interrelated variables are considered. Capturing these cross-variate dependencies demands sophisticated models capable of learning complex relations. Furthermore, the availability and quality of data is of great importance. Missing data, outliers, or insufficient historical records can influence the performance of forecasting models significantly. Finally, external factors such as economic shifts, policy changes, or unexpected events

might introduce unpredictability and uncertainty, causing difficulty in delivering accurate forecasts. These challenges highlight the need for robust and adaptive forecasting models, particularly in domains like dairy forecasting, where accurate predictions are important for decision-making.

0.1.10 Evaluation criteria

The most common evaluation criteria used to assess the performance of dairy forecasting models is the Mean Square Error (MSE), Mean Absolute Error (MAE), Rooted Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) calculated based on the actual and predicted profits (Hyndman et Koehler, 2006). RMSE can be calculated for each age in month separately. Therefore, we can interpret the RMSE for each month as :

$$RMSE_m = \sqrt{\frac{1}{N_{cows}} \sum_{c \epsilon cows} (\hat{p}_{c,m} - p_{c,m})^2} \tag{15}$$

Where $p_{c,m}$ and $\hat{p}_{c,m}$ represent the actual and predicted profit (income) values for the c'th cow in month m. After computing the RMSE for each month, the final RMSE (total RMSE) to evaluate the model is calculated using the following formula :

$$RMSE_{total} = \sqrt{\frac{1}{N_{cows} \times N_{months}} \sum_{m \epsilon months} \sum_{c \epsilon cows} (\hat{p}_{c,m} - p_{c,m})^2} \tag{16}$$

Where, N_{months} typically is 11 or 12 as we predict the income or profits along one lactation period. A lower RMSE indicates a more accurate forecast of cow profitability.

There are many assessment criteria in time series forecasting field, which have been used in the literature to verify the prediction power of various methods.

Another criteria for assessing the time series prediction results is the Mean Absolute Error (MAE) criteria which is defined as follows (in terms of dairy milk forecasting) :

$$MAE_{total} = \frac{1}{N_{cows} \times N_{months}} \sum_{m \epsilon months} \sum_{c \epsilon cows} |\hat{p}_{c,m} - p_{c,m}| \tag{17}$$

Third evaluation criteria is the Mean Absolute Percentage Error (MAPE) :

$$MAPE_{total} = \frac{1}{N_{cows} \times N_{months}} \sum_{m \epsilon months} \sum_{c \epsilon cows} \frac{|\hat{p}_{c,m} - p_{c,m}|}{p_{c,m}} \times 100\% \tag{18}$$

In the context of time series forecasting, bias is another metric used to evaluate forecasting accuracy. However, a highly biased prediction indicates a significant issue in the forecasting process.

$$Bias_{total} = \frac{1}{N_{cows} \times N_{months}} \sum_{m \epsilon months} \sum_{c \epsilon cows} (\hat{p}_{c,m} - p_{c,m}) \tag{19}$$

The prediction results of the proposed models and the other baselines are evaluated and compared in terms of different error metrics. It should be mentioned that the best performing model is specified as the one with the lowest value of the above mentioned evaluation metrics.

0.2 Thesis overview

The primary objective of this thesis is to develop a model enhanced for time series forecasting of dairy production. This is achieved through the designed models to capture temporal and intricate cross-channel correlations. To address the challenges of time series forecasting, particularly in dairy field, two deep learning models are proposed for time series forecasting, which represent the critical cross-channel and temporal dependencies, effectively. During the course of my research, I developed a multiscale model for multivariate time series forecasting that accounts for the inherent multi-scale patterns present in time series data (Naghashi *et al.*, 2025a). This method utilizes Transformer encoders to capture temporal and cross-channel dependencies. Moreover, a multi-scale time series embedding method is innovated which projects multiple scales of the time series into a model space by using 1-dimensional convolutions. In the second designed model, a recurrent neural network model is proposed which leverages Gated Recurrent Units (GRU) to capture intricate cross-channel dependencies and integrates a feed-word layer to address temporal patterns (Naghashi *et al.*, 2025b). We further extended this model into a bidirectional iGRU model, which operates in

both directions along the channel dimension to capture complementary dependencies. This approach outperforms most of the state-of-the-art models, delivering accurate prediction results for dairy production forecasting across different lactation periods.

0.2.1 Thesis structure

This thesis offers a comprehensive exploration of time series forecasting, with a focus on its application to dairy production and the associated research contributions. It begins with an overview of time series forecasting models, establishing the context for addressing the specific challenges of dairy forecasting. The research problems and hypotheses are defined within the broader field, providing a clear foundation for the study. A detailed review of state-of-the-art time series forecasting models, particularly those tailored to dairy prediction, illuminates current methodologies and their limitations. The primary research contributions are then explained in depth, highlighting their significance and contributions. Finally, the conclusion evaluates these contributions and proposes directions for future research.

- Chapter 1 : Background
- Chapter 2 : Benchmark datasets
- Chapter 3 : Research problems
- Chapter 4 : Related work
- Chapter 5-9 : Research contributions in detail
- Chapter 10 : Conclusion

The research projects in this thesis are inter-related. Three innovative multivariate time series forecasting models (Transformer-BEATS, MultiPatchFormer and iGRU) are proposed, which forecast time series associated with different fields, including traffic, energy, electricity consumption, and finance (Wu *et al.*, 2021). We also designed an extended version of the iGRU model (Bi-iGRU) for prediction of cumulative milk income and other dairy factors across different lactation periods, which delivers promising results. More specifically, Bi-iGRU captures cross-channel dependencies (relations between multiple dairy series) in two directions which is proved to gain more information compared to the uni-directional setting according to the conducted ablation experiments. Furthermore, this model provides forecasts for different dairy variables, including cumulative and daily milk income. This thesis includes the contents of published and submitted papers which represent its main contributions :

— Chapter 4 : Naghashi, Vahid, Gabriel Dallago, Abdoulaye Banire Diallo, and Mounir Boukadoum. "Univariate and multivariate time-series methods to forecast dairy income." In 2nd AAAI Workshop on AI for Agriculture and Food Systems. 2023.

— Chapter 5 : Naghashi, Vahid, Mounir Boukadoum, and Abdoulaye Banire Diallo. "Transformer-BEATS : A Transformer model for time series prediction of dairy milk production." In ISCB-LATAM SoIBio CCB-COL (2024), Poster.

— Chapter 6 : Naghashi, Vahid, Mounir Boukadoum, and Abdoulaye Banire Diallo. "A multiscale model for multivariate time series forecasting." Scientific Reports 15, no. 1 (2025) : 1565.

— Chapter 7 : Naghashi, Vahid, Mounir Boukadoum, and Abdoulaye Banire Diallo. "Should We Reconsider RNNs for Time-Series Forecasting ?." AI 6, no. 5 (2025) : 90

— Chapter 8 : Naghashi, Vahid, Mounir Boukadoum, and Abdoulaye Banire Diallo. "A Bi-directional cross-channel RNN model for time-series forecasting of dairy production." Scientific Reports 15, Article number : 44967 (2025).

To facilitate understanding of the progression of ideas across the different contributions, Table 0.1 provides a synthetic overview of the proposed models in this thesis. The table highlights the core modeling principles, and the key improvements introduced at each section. This synthesis makes explicit how each contribution builds upon the previous ones to progressively address the challenges inherent to multivariate time-series forecasting in dairy production.

Table 0.1 – Simplified synthesis of the contributions proposed in this thesis, highlighting the core modeling intuition and the progression of ideas across chapters.

Contribution	**Intuition**	**Improvement Over Previous Contribution**
Chap. 4 (MuMu+Attention)	Introduces a contextual attention mechanism on top of stacked LSTMs to allow the model to focus on biologically relevant periods of the lactation cycle rather than relying solely on the final hidden state.	Provides a meaningful aggregation of temporal information compared to standard RNNs by weighting informative time steps.
Chap. 5 (Transformer-BEATS)	Uses self-attention to model long-range temporal dependencies and inter-variable relations in parallel, overcoming the sequential limitations of recurrent architectures.	Utilized self-attention to capture long-term dependencies and cross-channel correlations explicitely beyond LSTM-based models.
Chap. 6 (MultiPatchFormer)	Explicitly models seasonality and temporal dynamics at multiple resolutions through multi-scale embeddings, captures cross-channel dependencies and uses semi-autoregressive decoding to reduce noise.	Extends the Transformer-based approach by addressing variable-length seasonal patterns that single-scale attention cannot capture effectively.
Chap. 7 (iGRU)	Reconsiders RNNs by applying GRU cells along the variable (channel) dimension to directly model inter-variable dependencies in an efficient and structured manner.	Shifts the focus from temporal attention to explicit cross-channel modeling while significantly reducing computational complexity compared to Transformers and captures temporal dependencies implicitely though the feed-forward layer.
Chap. 8 (Bi-directional Cross-Channel RNN)	Introduces bidirectional processing across variables to capture directional and reciprocal interactions between physiological and management factors.	Completes the cross-channel modeling introduced in the previous chapter by capturing interactions in both directions, leading to more expressive inter-variable representations.

CHAPITRE 1
BENCHMARK DATASETS

The most comprehensive data on food supply and consumption are provided by the United Nations Food and Agriculture Organization (FAO). According to this information, per capita calorie supply has increased from 1961 to 2015 (Bruinsma *et al.*, 2012). This increase in food consumption is the most crucial reason to improve agricultural methods. Suggesting and bringing the best precision tools for farmers helps them in the decision making process. Here, the main challenges include helping the producers keep track of their farms and ameliorate their efficiency. The dairy dataset includes data files covering supply, demand and trade of various dairy products, such as milk, skin and meat. Dairy temporal data usually represent a set of measurements related to animals (cows) captured in the test dates over a period of time, for example, over a sequence of days or consecutive months. In this work, the dairy data is collected by Lactanet during the years of 2006 to 2017 from 6675 herds and associated with 1482383 cows. All these cows correspond to dairy farms in Quebec, Canada. Each line in the dataset corresponds to a test date and is identified by a cow ID. Therefore, each line consists of different feature values (factors) measured on a specific test day for a cow with the specified ID. A refined version of the dataset contains 15 various factors (Frasco *et al.*, 2020) and these features are illustrated in Table 1.1, along with their corresponding missing data rates. Some of the cows that left the herd between months 18 and 46 after their birth were removed from the dataset. In addition, cows without information on their milk value factor were eliminated. In this work, only cows of the Holstein breed have been taken into account for the purpose of dairy prediction.

The dairy production dataset can be defined as a multivariate time series data composed of multiple dairy-related factors. Following standard practices in dairy data collection (Schaeffer et Jamrozik, 1996), our dataset utilizes a single-monthly-sample-per-cow policy. Specifically, features are measured on one or two test days in each month, with each cow sample for a specific month represented as a row in the dataset (Frasco *et al.*, 2020).

1.1 Life-time profit

Life-time profit of a cow is defined as the difference between the cumulative value of the milk yield and the cost associated with the cow's management and feeding. Several factors impact the profits and income of dairy production. These factors are collected through connected sensors, interactive dashboards, and

Table 1.1 – Features used in the prediction of dairy profits

Feature	Missing Rate
BHB	88.93%
CCD	0%
DIM	35.4%
HR_24_FT	35.4%
HR_24_MILK	35.4%
HR_24_PT	35.4%
LACTOSE	46.81%
LACT_NO	0%
MLKNG_FQCY	0%
MLKNG_PTRN	0%
MUN	56.53%
Month	0%
SCC	36.35%
Year	0%
Daily_Profit	35.53%

questionnaires. Many of these features are interrelated. For example, environmental conditions directly affect production but can also impact the health, management practices, and genetic characteristics of a cow, all of which indirectly influence dairy production. In fact, there are other factors, which impact the production profit, such as the state of the product market and behaviours of a farmer, which are beyond the scope of this study. Figure 1.1 illustrates the different types of variables found in dairy production, where dotted lines show one-way interactions between the variables, dashed lines mutual interactions, and solid lines correspond to the relations selected to be modeled in MuMu-RNN model (Frasco *et al.*, 2020) and our work. Estimation of the life-time income of cows helps farmers to monitor their farms more efficiently and reduce the related costs by using resources efficiently.

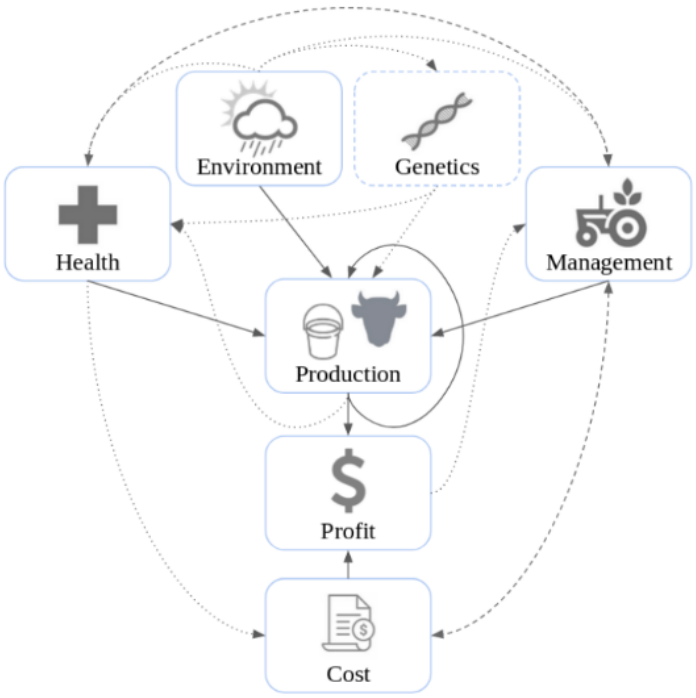


Figure 1.1 – The interactions between different factors in dairy production.

1.2 Problems studied with the collected dairy data

Milk production profit is calculated for each cow sample according to amount of the milk it produces and the associated feed costs. (Frasco *et al.*, 2020) applied a deep learning based model to forecast the dairy profits in the future months. In that work, two univariate and multivariate models are developed for profit forecasting. The first model known as Uni-RNN (Frasco *et al.*, 2020), takes the milk profits in the past months (early lactation periods) and provides the future profits as its output. On the other hand, their second model (MuMu-RNN) utilizes all the factors, including early dairy profits, and predicts the profits along the next lactation period. In their study, Holstein cows were used, and several preprocessing steps were applied prior to model training. For example, cows that left the herd shortly after birth were excluded from the dataset to ensure data consistency. Moreover, two ordinal features—month and milking frequency—were converted into their one-hot encoded representations. A new feature, Profit, was computed by subtracting the daily cost of maintaining a cow from the daily value of the milk it produced. Missing values were handled using various strategies, including imputation with the herd-level mean of the corresponding feature and predefined baseline values to fill in missing data values.

1.3 Dairy factors and indicators

Dairy factors include health indicators, environmental conditions, milk quality factors and seasonal indicators. In this section, we will go through the definition of different factors and their relation with dairy income.

1.3.1 Health indicators

The features related to cow health show the presence of a disease or disorder in dairy cattle by measuring a specific bio-chemical in blood or milk sample of a cow. Health factors studied in most of the research papers and projects include the following factors :

1) BHB (Beta-Hydroxybutyrat) : Hyperketonemia (HYK) is one of the most frequent and costly metabolic disorders in dairy cows and its diagnosis is based on measuring BHB concentration in blood samples (Benedet *et al.*, 2019). Hyperketonemia has been related to greater milk fat content, fat-to-protein ratio, energy-corrected milk, and lower protein and urea nitrogen in milk (Benedet *et al.*, 2019). Therefore, BHB and Fat-to-Protein ratio are highly correlated factors. It is concluded that BHB and milk profitability (based on the produced milk) are negatively correlated. In general, HYK (High BHB) increases the risk of health issues during early lactation, with consequent negative effects on herd profitability (Benedet *et al.*, 2019).

2) Somatic Cell Count (SCC) is a key indicator of milk hygiene, as it reflects the health status of the cow's udder. According to (EL-TAHAWY et EL-FAR, 2010), an elevated SCC is associated with reduced milk yield and lower profitability, indicating an inverse relationship.

3) CCD (Condition Code) : CCD is a health indicator related to cow illness. CCD value of 1 means the cow is sick and milk production will be affected (e.g. decrease in milk production).

1.3.2 Management and milk indicators

Lactation refers to the period during which a cow produces milk following calving, usually lasting around 305 days under standard dairy management. This is followed by the dry period, a non-lactating phase, usually lasting 45 to 60 days, allowing the cow's udder tissue to regenerate before the next calving phase. Proper management of both phases is essential for optimizing milk production, fertility, and overall herd health.

The management factors are described as follows :

1) DIM : Days in Milk (DIM) is closely associated with the duration of the dry period and serves as a useful indicator of reproductive efficiency and overall herd management. Ideally, the 12-month rolling average for DIM should fall within the range of 160 to 170 days. An average DIM greater than 200 indicates a reproductive problem, such as delayed conception or extended calving intervals. A large DIM value results in lower lifetime milk production per cow.

2) $HR.24.FT$: Fat yield (kg) recorded during the 24-hour test period. It directly contributes to milk value (income) and reflects the fat production performance of the cow on the test day.

3) $HR.24.PT$: Protein yield (kg) recorded on the test day. It directly reflects the cow's protein production performance and is a key indicator of milk quality and market value.

4) $HR.24.MILK$: Milk yield (kg) recorded during the 24-hour test period. It serves as a direct measure of the cow's daily milk production performance.

5) Milk Urea Nitrogen (MUN) : As a management tool for dairy farmers, MUN offers a simple approach to examine protein status of rations fed to dairy cattle (Kohn, 2007). Through routine monitoring of MUN, dairy farmers can adjust dietary protein levels to better match protein requirements of their cows and potentially increase profitability by reducing feed costs. High values typically show inefficient protein utilization in dairy cattle which might lead to low milk production.

6) Lactose : According to Lactanet standards, the lactose percentage measured on the test day is directly related with milk yield. Higher lactose concentrations promote an osmotic influx of water into the mammary gland, which in turn increases milk volume (Erickson et Kalscheur, 2020).

7) $LACT.NO$ (Lactation Number) : Milk yield differs in each lactation period and lactation number is a meaningful indicator of milk yield. According to (Vijayakumar *et al.*, 2017) there is a considerable relationship between the first, second lactation and milk yield, whereas the maximum milk yield is usually produced in the fourth lactation.

8) $MLKNG.FQCY$: Milking frequency shows the frequency of milking on a test day and is highly related

to milk yield and profitability.

9) $MLKNG.PTRN$: Milking pattern categorizes the milking schedule or timing for each cow. The codes and their corresponding meanings are as follows :

- 00 – UNKNWN : Unknown milking pattern ; data not recorded or unavailable.
- 01 – 24 HR : Milking occurred over a 24-hour period ; possibly indicative of automated milking systems (AMS) where cows are milked at various times throughout the day.
- 02 – AM : Milking conducted in the morning.
- 03 – PM : Milking conducted in the evening.
- 04 – Other : Milking pattern does not fit into the above categories ; could include irregular or alternative milking schedules.

10) DAY, MONTH, YEAR : Day, month and year of a milking test is strongly correlated with milk yield since the milk production is affected by season and test date.

1.4 Dairy dataset associated with 3 lactation periods

We created a multivariate time series dataset containing a set of dairy attributes, including metrics of milk quality, seasonality, year, health, and management factors, recorded during the first, second, and third lactation of 147,749 dairy cows from 5,844 Canadian dairy farms over the years of 2006 to 2017 by Lactanet (a leading organization in Canada responsible for the national genetic evaluation and dairy herd management services). The prediction targets are the monthly income from milk sales (in Canadian Dollar) for each cow during their third lactation. To standardize the time series across animals, the lactation length was fixed at 11 months for the first, second, and third lactations. This duration was chosen based on the mean plus one standard deviation of lactation lengths observed in the training cow samples. For cows with lactations shorter than 11 months, additional data rows were introduced for the missing months and imputed using linear interpolation between the two closest available time points. The following steps were taken to clean the data : 1- Keeping only animals having test records in the first, second and third lactations, 2- Deleting the records from the dry period : dry period is defined as the months in which a cow doesn't produce milk (which mostly occurs between two lactation cycles) and milk value (income) is almost zero. Since there is no income from sales during the dry period, records associated with these months were deleted, 3- Removing the animals who left their herd before and during the third lactation, 4- Deleting duplicate records (only one test kept for each month), 5- Deleting the records with negative milk value and negative cumulative

milk value : The reason for removing such records, which constituted a small percentage of the data, was to remove the inconsistencies in the data set, as the milk value is the income from milk sale (CAD) and negative values of dairy income could be accounted for an inconsistency due to the errors in data acquisition steps, 6- Deleting the records which included contradictions, e.g., rows indicating the cow was milking, while the milk yield of those records was equal to zero, 7- Outlier removal (their corresponding rows) : Values falling outside the range of the mean ± 2.5 times the standard deviation for a given dairy attribute were identified as outliers and subsequently removed from the dataset. All the dairy attributes used as input to our model and their descriptions are shown in Tables 1.2 and 1.3.

Table 1.2 – Numerical dairy variables used in our first version of dairy dataset.

Variable	Missing %	Min	Max	Mean	Std
Milk produced in 24 hours (Kg)	1.4	0.20	61.40	29.60	9.25
Milk income (CAD)	0.002	0.09	42.95	21.21	5.94
Somatic Cell Count (1000/mL)	17.62	1	1754	156.59	254.87
Milk Urea Nitrogen	56.04	2.10	19.80	10.91	3.27
Lactose yield (Kg)	37.63	0	6.30	4.55	0.22
Fat yield in 24 hours (Kg)	1.64	0	2.40	1.19	0.38
Protein yield in 24 hours (Kg)	1.64	0	1.94	1.00	0.30
Number of days cow has been in milking	0	0	517	169.56	99.39
Fat to protein ratio	1.64	0.73	1.67	1.20	0.16

Table 1.3 – Categorical dairy variables used in the first version of the dataset.

Variable	Missing %	Levels (% of observations)	Total Observations
Lactation Number	0	1 : 33.33%, 2 : 33.33%, 3 : 33.33%	4,875,717
Milking/Day	0	1 : 3.33%, 2 : 95.37%, 3 : 1.28%	4,875,717
Test Season	0	1–4 (Each 25%)	4,875,717
Birth Season	0	1 : 22.75%, 2 : 27.18%, 3 : 26.14%, 4 : 23.91%	4,875,717
Animal Condition	0	1 : 99.40%, 4 : 0.06%	4,875,717
Test Year	0	6–17 (Each 3–10.7%)	4,875,717

1.5 Dairy dataset associated with different lactation periods

In the extended version of the dairy dataset, only animals that survived at least until the end of the sixth lactation period are included. As mentioned in the description of the previous dataset, each test record

consists of various features obtained from milk samples and management-related measurements. We pre-processed the dataset by imputing missing values, removing outliers, and deleting the inconsistent data records. To ensure consistency and completeness, duplicate and inconsistent records are removed. For instance, records with negative milk income or negative 24 hour milk production are deleted. Additionally, we removed records associated with the dry period of animals during their lactation periods.

For missing data imputation in the dataset, we employed a three-step approach. First, we grouped animals by herd ID, test year, and season, imputing missing values within each group using the mean of each group where the animal corresponds. Next, for any remaining missing values, we imputed them using the mean of the associated herd, based on herd ID. Finally, we addressed any still-unresolved missing values by imputing them with the mean of groups defined by test year alone. The aforementioned three-step imputation approach leverages the hierarchical structure of the dairy dataset to guarantee accurate and contextually relevant replacements for missing values. These steps minimizes bias by prioritizing the most specific available data while progressively considering broader replacements. At the end, 19 relevant dairy variables are selected and constructed a multivariate time series based on their recorded variable sequences over six lactation periods (72 months). The selected dairy variables include, daily milk income, cumulative milk income, milking pattern, milking frequency, ANS-CD (animal status code), lactation number, days in milk, milk produced in 24 hours (kg), 24-hour fat, 24-hour protein, lactose, abnormal status, Somatic cell count (SCC), Milk Urea Nitrogen (MUN), test year, test month, test day, test season and condition code (a sickness indicator). Description of the selected dairy variables are shown in Table 1.4. In addition, we converted the categorical variables into their one-hot encoded version in order to leverage their corresponding information, efficiently. The continuous variables are normalized using mean and standard deviation. The distribution of continuous dairy variables after normalization is illustrated in Figure 1.2. As this Figure illustrates, all the variables follow a symmetric distribution, while Lactose has some extreme values. Finally, we obtained 97 features associated with the time series channels. In our initial assessment of the dairy dataset, we examined the correlations between variables such as milk production, health indicators, milk quality and seasonality factors. This correlation analysis, visualized in Figure 1.3, supports our hypothesis that cross-channel dependencies are critical for effective time series forecasting of dairy income. We will discuss more about forecasting steps in Chapter 8.

To evaluate model performance, the dataset is randomly divided into training and test sets in an 80 :20 ratio. We created six different versions of the dataset, each corresponding to a different lactation period

Table 1.4 – Description of dairy variables used to forecast milk income.

Variable	Description	Min	Max
Daily Milk Value	Income from milk produced in the test day	0	74.59
Cumulative Milk Value	Cumulative milk income along the lactation months	0	49477
Milking Pattern	Milking pattern of an animal	1	4
Milking Frequency	The milking frequency of the animal for the test (1 time per day, 2 times per day)	0	3
Animal Status Code	Animal status indicator (1 indicates dryness, 2 indicates milking)	1	4
Lactation Number	Lactation number in {1, 2, 3, 4, 5, 6}	1	6
Days in Milk	Number of days that a cow has been milking along the lactation months	0	1575
24-hour-Milk	Milk produced (Kg) in 24 hours	0	90.9
24-hour-Fat	Fat in milk produced in 24 hours	0	5.802
24-hour-Protein	Protein in milk produced in 24 hours	0	3.342
Lactose	Lactose percentage in milk sample (24 hours)	0	7.25
Abnormal Status	Quality of the sampled data on the test day	0	8
Somatic Cell Count (SCC)	Somatic cell count in the milk sample	1	25339
Milk Urea Nitrogen (MUN)	Milk Urea Nitrogen (mg / dL)	0	39.9
Test Year	Year indicator of the test date	6	17
Test Month	Month of the test date	1	12
Test Day	Day of the test date	1	31
Test Season	Season of the test	1	4
Condition Code	Code indicator of the animal condition (0 healthy, 1 sick)	0	1

treated as the forecast horizon. In each case, the input consists of records from one or more past lactation periods, while the output represents the upcoming lactation period. For instance, in the first version, only the first lactation is used as input, and the second lactation serves as the forecast horizon. In the second version, the input includes data from the first and second lactation periods (covering 24 months), while the third lactation (months 25 to 36) is the forecast horizon. The other four versions follow this pattern, progressively incorporating more lactation periods as input and the next lactation as the forecast window (12 months).

1.6 Public time series datasets

We assessed the performance of our time series forecasting models using established public benchmarks. These datasets, widely recognized in the field of time series analysis, span diverse applications such as meteorology, finance, traffic analysis, electricity demand, and energy forecasting. We shortly decribe them

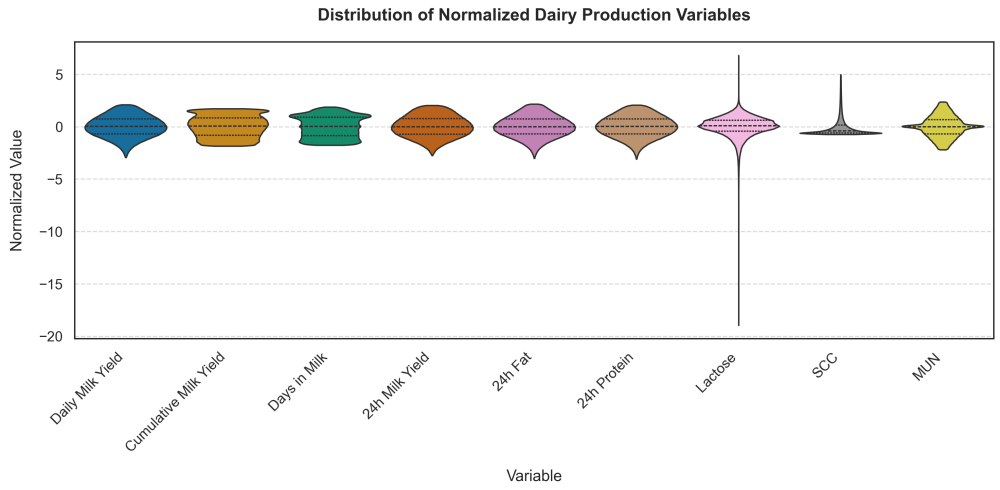


Figure 1.2 – Violin plot of the selected dairy variables after normalization by using their mean and standard deviation.

here and refer the interested reader to their references for further details (Wu *et al.*, 2021). The Traffic dataset (Wu *et al.*, 2021) is a collection of road occupancy data from the California Department of Transportation. It was gathered from 862 sensors between 2015 and 2016, capturing the percentage of road occupancy (traffic flow) at various locations. This dataset is widely used for evaluating forecasting models in transportation systems due to its high dimensionality (many variates) and temporal dependencies. The PEMS dataset (Liu *et al.*, 2022) is a complex spatio-temporal dataset related to public traffic networks in California, consisting of four subsets (PEMS03, PEMS04, PEMS07, PEMS08). Each subset contains traffic flow, occupancy, and speed data recorded at 5-minute intervals, providing a valuable resource for studying spatio-temporal forecasting challenges. The ETT (Electricity Transformer Temperature) dataset (Zhou *et al.*, 2021) includes data related to the load and oil temperature of electricity transformers, collected from July 2016 to July 2018. This collection contains multiple datasets captured at hourly scale (ETTh1, ETTh2) and minute-level (ETTm1, ETTm2) granularity, each consisting of seven variates. The ETT dataset is particularly useful for studying long-term dependencies and multi-variate forecasting in energy systems. The Weather dataset (Wu *et al.*, 2021) consists of 21 meteorological variates, such as temperature, humidity, and wind speed, recorded every 10 minutes from the Max Planck Institute for Biogeochemistry. Its high-frequency sampling and diverse features make it a benchmark for evaluating forecasting models in environmental and climate studies. The Electricity dataset (Wu *et al.*, 2021) contains hourly electricity consumption records of 321 customers over several years. It is commonly used to evaluate forecasting models in the energy domain due to its strong seasonality and variability in consumption patterns. The Solar-Energy dataset (Wu *et al.*,

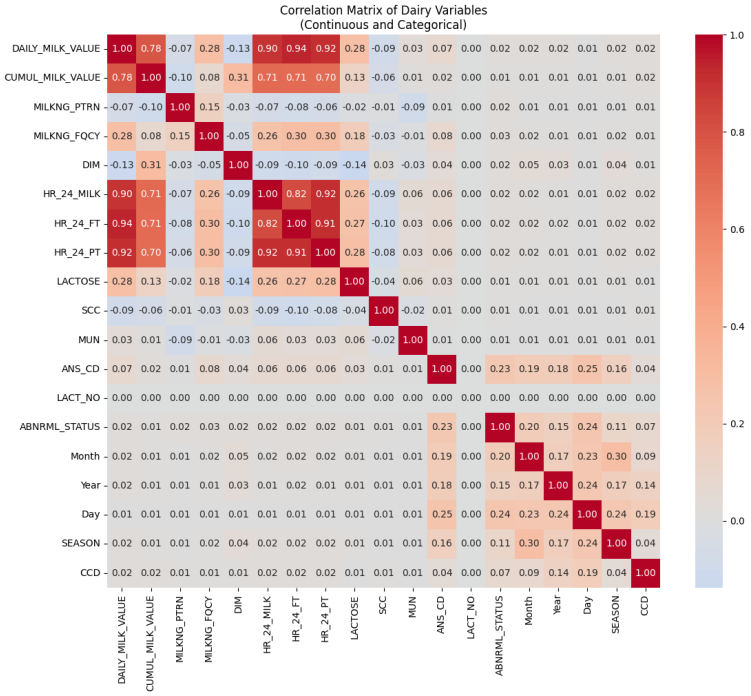


Figure 1.3 – An illustration of the correlations between different dairy variables.

2021), sampled every 10 minutes, contains solar power generation records from 137 photovoltaic (PV) plants in Alabama during 2006. This dataset is useful for studying renewable energy forecasting, as it captures the intermittent nature of solar power generation. Finally, the Exchange dataset (Wu *et al.*, 2021) is based on panel data of daily exchange rates for eight countries from years 1990 to 2016. It is widely used for financial time series forecasting, as it exhibits complex dynamics influenced by global economic factors. More details regarding the aforementioned datasets are shown in Table 1.5.

Table 1.5 – Dataset Information : Dim represents the number of variates and Dataset Size denotes the total number of time points in (Train, Validation, Test) split of each dataset, respectively. Prediction Length indicates the future time points to be predicted, and four prediction lengths settings are specified in each dataset. Frequency denotes the sampling interval of time points.

Dataset	**Dim**	**Prediction Length**	**Dataset Size**	**Frequency**	**Information**
ETTh1, ETTh2	7	{96, 192, 336, 720}	(8545, 2881, 2881)	Hourly	Electricity
ETTm1, ETTm2	7	{96, 192, 336, 720}	(34465, 11521, 11521)	15min	Electricity
Exchange	8	{96, 192, 336, 720}	(5120, 665, 1422)	Daily	Economy
Weather	21	{96, 192, 336, 720}	(36792, 5271, 10540)	10min	Weather
Electricity	321	{96, 192, 336, 720}	(18317, 2633, 5261)	Hourly	Electricity
Traffic	862	{96, 192, 336, 720}	(12185, 1757, 3509)	Hourly	Transportation
Solar-Energy	137	{96, 192, 336, 720}	(36601, 5161, 10417)	10min	Energy
PEMS03	358	{12, 24, 48, 96}	(15617, 5135, 5135)	5min	Transportation
PEMS04	307	{12, 24, 48, 96}	(10172, 3375, 3375)	5min	Transportation
PEMS07	883	{12, 24, 48, 96}	(16911, 5622, 5622)	5min	Transportation
PEMS08	170	{12, 24, 48, 96}	(10690, 3548, 3548)	5min	Transportation

CHAPITRE 2

PROBLEM STATEMENT, RESEARCH QUESTION AND HYPOTHESES

2.1 Problem statement

In this thesis, we aim to provide an automatic framework for farmers to manage their herds efficiently. This framework is based on the prediction of future milk production. The problem we deal with is defined as follows : Given the features in the early test dates (early lactation periods) of a cow, the future milk production is predicted over the upcoming months. Therefore, the input to our problem is a sequence of factors related to the health, environment and milk quality of a cow in the early test months. Our problem can be treated as a multivariate time series forecasting task, which includes several series of dairy characteristics. The length of the input sequence is T, which corresponds to the dairy information of animals along the early lactation periods. According to Figure 2.1, in one instance of the posed problem, the early months are related to the first and second lactation and the late months correspond to the third lactation period.

The milk production forecasting problem can be formulated as follows : Given a herd of N cows, each cow i is associated with a set of multivariate time series records

$$\left\{\mathbf{x}_j^{(i)} = (x_{j,1}^{(i)}, \ldots, x_{j,T}^{(i)})\right\}_{j=1}^{v} \in \mathbb{R}^{v \times T},$$

where v denotes the number of dairy variables (features), and T is the fixed length of the early lactation periods recorded for all animals. Each $\mathbf{x}_j^{(i)}$ represents the temporal evolution of the j-th dairy variable for cow i.

The objective is to predict M future values of milk income during the next lactation period for each cow :

$$\hat{\mathbf{r}}^{(i)} \in \mathbb{R}^M.$$

Therefore, the goal is to learn a nonlinear function f that maps the input multivariate time series

$$X \in \mathbb{R}^{N \times T \times v}$$

to the target outputs

$$\hat{R} \in \mathbb{R}^{N \times M},$$

where X is the training set consisting of all N cow samples.

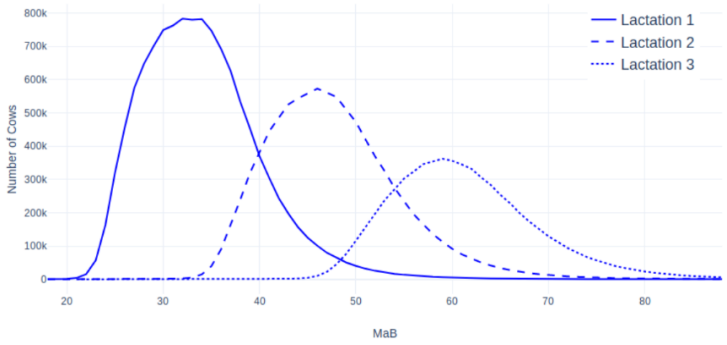


Figure 2.1 – Different lactation curves (Early and late lactation) (Frasco *et al.*, 2020)

To be more specific, the prediction model is first trained by providing the training dairy samples to it. The training process is repeated for several iterations (training epochs) and after convergence, the trained model is applied to the test samples for the purpose of forecasting their related future income. In this thesis, the target series represents the daily or cumulative milk income over a sequence of months. The forecasting problem can be treated in two following distinct scenarios. In the first case, dairy income from previous lactation periods (past months) is modeled as a univariate time series, where only historical income data serves as input to the model. This formulation defines the problem as a univariate time series forecasting task. In the second scenario, multiple dairy-related characteristics are used as input variables in the past months (a multivariate time series). The predicted output remains the same as in the univariate case, where only the dairy income is predicted, or alternatively, multiple dependent variables can be predicted simultaneously, forming a multivariate forecasting type of problem. This approach allows for modeling of complex interactions among features, potentially improving forecasting accuracy and robustness. In the multivariate case, all of the series have the same length and their length usually depends on the problem at hand. In many forecasting problems the context size of the input series is selected as a multiplication of the forecasting horizon (the number of steps supposed to be predicted) (Oreshkin *et al.*, 2019).

Univariate case Given a length-H forecast horizon, the length of observed series history is chosen to be 2H, 3H, ... and the task is to predict the future milk production $\mathbf{r} \, \epsilon \, \Re^H = [r_{T+1}, r_{T+2}, ..., r_{T+H}]$ based on an observed series of length T, $\mathbf{r^{early}} \, \epsilon \, \Re^T = [r_1, r_2, ..., r_T]$.

Multi-variate case. Given an observed dairy production series along with observed values of other dairy-related factors across the past months, the goal is to predict dairy profitability for future lactation periods. Specifically, we aim to predict future dairy production values $\mathbf{r} \in \mathbb{R}^H = [r_{T+1}, r_{T+2}, \ldots, r_{T+H}]$ using

spatial and temporal information from multiple time series of length T. The observed time series $\mathbf{X} \in \mathbb{R}^{f \times T} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T]$ corresponds to all input dairy features or indicators, where $\mathbf{x}_t \in \mathbb{R}^f$ represents the feature values at the t-th time step (month) :

$$\mathbf{x}_t = \begin{bmatrix} x_t(1) \\ x_t(2) \\ \vdots \\ x_t(f) \end{bmatrix}.$$

2.2 Challenges and motivation

As previously discussed, dairy income is influenced by a range of factors, including environmental conditions, animal health status, and genetic background. Incorporating all these variables can significantly enhance the accuracy of profit forecasting. However, measuring and capturing this diverse information is both costly and time-consuming. Furthermore, these factors are often unavailable at certain time steps due to sensor failures or inconsistent data acquisition. Consequently, the dairy dataset contains a substantial number of missing values that must be carefully imputed using meaningful and context-aware methods. Therefore, addressing missing data is a critical challenge in developing a reliable forecasting system.

Another major challenge lies in effectively capturing both temporal dependencies (relationships between past and future values of the same variable) and cross-channel correlations (interactions between different variables). An accurate prediction model must utilize both types of dependencies to improve performance.

In addition, appropriate pre-processing steps are required to ensure data quality for model training and validation. These steps typically include missing value imputation, normalization of continuous features, and one-hot encoding of categorical or ordinal variables to extract meaningful information from them.

However, the main challenge is the selection or design of an efficient predictive model that can learn and capture complex patterns within the data. The target dairy income sequences in our dataset exhibit some fluctuations and irregular patterns, which make accurate forecasting difficult. Moreover, real-world time series is composed of multiple granularities and models that incorporate multi-scale representations and capture cross-channel dynamics tend to perform efficiently in these complex forecasting tasks.

2.3 General research question

Based on the concepts introduced previously, we propose the following research question : Given a set f of dairy related factors captured during a sequence of T months, how can we predict late milk production in the upcoming months with high accuracy ?

To address this question, we will tackle the following sub-questions in this thesis.

2.3.0.1 **Research sub-questions**

Q1. How to model lifetime milk production and design a robust deep learning architecture to handle the challenges associated with temporal dairy data ?

Q2. How can we better capture long-term and cross-channel dependencies in the spatio-temporal dairy dataset ?

Q3. How can missing values in the dataset be effectively handled to enhance data integrity and improve model performance ?

2.4 Research objectives

To answer the sub-questions, we propose the following objectives :

Obj1. Assess several approaches to model lifetime milk production and design an original deep learning model that provides the most accurate predictions. The model should capture temporal and cross-channel information from the input time series.

Obj2. Design and evaluate several architectures to capture temporal and cross-variate dependencies according to characteristics of the data and the domain knowledge.

Obj3. Design a method to handle missing data in the context of lifetime dairy estimation.

To address these objectives, in this thesis, we will rely on the following hypotheses :

Hypothesis 1 : Future milk production during a given lactation period is affected by multiple factors observed in the preceding months, including environmental conditions, management practices, health indicators, and milk quality metrics. In addition, the production results of previous lactation(s) provide valuable predictive information for forecasting future yields.

Hypothesis 2 : It is plausible to achieve high predictive accuracy by first imputing the missing values in the dataset and applying deep learning models to the cleaned data afterwards.

Hypothesis 3 : Temporal feature selection methods, such as the attention mechanism (Vaswani *et al.*, 2017), enable the model to efficiently assign weights and importance to different time steps in the input sequence.

Hypothesis 4 : Not all dairy features are equally relevant or influential in forecasting milk production and there are interactions between time series channels (variates).

Hypothesis 5 : Real-life time series are composed of multiple time resolutions, which should be represented by the prediction model.

CHAPITRE 3
LITERATURE REVIEW

3.1 Introduction

In recent years, the dairy industry has witnessed a proliferation of deep learning-based approaches. For example, the long-short-term memory (LSTM) neural network has been used to predict cow's milk profitability (Frasco *et al.*, 2020), which relies on integrated information from multiple sources to predict profits in future months. A deep learning framework is proposed in (Liseune *et al.*, 2021) to encode the dairy features, then predict the latent representation of milk yield sequence and generate the lactation curve by leveraging Convolutional Neural Networks (CNN) and encoder-decoder modules. The prediction of the first test day milk yield is accomplished through the application of classical machine learning models, including the random forest in (Salamone *et al.*, 2022). This is achieved by giving the models information of test day production, reproduction and herd features. Models for time series forecasting are generally categorized into conventional statistical methods and deep learning models. Statistical models, such as ARIMA (Auto-regressive Integrated Moving Average) show a good performance (Contreras *et al.*, 2003). ARIMA corresponds to a class of linear models and its main shortcoming is that it is not applicable to non-linear systems and is a univariate model, not capable of utilizing other covariates in the forecasting process (Zhang *et al.*, 1998). In this section, we will introduce both classical and deep learning based time series forecasting models. First, we will have a brief review of the applications of machine learning and deep learning in agriculture and dairy industry.

3.2 Application of deep learning and machine learning in agriculture

Application of deep learning methods in agriculture has increased in recent years. Many investigations have been conducted in the field of application of deep learning methods in agriculture, such as classification (Sun *et al.*, 2020), yield estimation (Rahnemoonfar et Sheppard, 2017), recognition and identification (Sladojevic *et al.*, 2016). Deep learning techniques have dealt with sequential and temporal agricultural data in different applications. (Song *et al.*, 2016) modeled spatio-temporal distribution of soil moisture using deep learning-based cellular automata. A multi-temporal deep learning approach based on Long-Short Term Memory neural network with attention mechanism is proposed for dynamic soy and soybean mapping (Xu *et al.*, 2020). Another application of DL in temporal agricultural data analysis is the prediction of water table depth

in agricultural areas (Zhang *et al.*, 2018).

Dairy farming is one of the largest sectors in agriculture that has been under study through data-driven methods. Promising results are obtained in automatic cropping of cow's body region and cow's pattern identification for individual animals (Zin *et al.*, 2018). Detection of the key parts of dairy cows (head, back and legs) is performed using deep convolutional neural networks and the YOLOv3 algorithm with high accuracy (Jiang *et al.*, 2019). Promising results are obtained for calving prediction from activity, lying and ruminating behaviors in dairy cattle (Borchers *et al.*, 2017). A machine learning framework for the early detection of a disease (ketosis) in dairy cows is proposed by using a Support Vector Machine applied to health and production factors (Ushikubo *et al.*, 2017).

3.3 Univariate time-series forecasitng methods

3.3.1 **Classical and statistical methods**

One of the well-known classical time series prediction models is the Auto-ARIMA (Contreras *et al.*, 2003). ARIMA corresponds to a class of models which utilizes the past values (lags and the lagged forecast error) to predict the future values. This model is based on a linear model adapted for seasonality and stationarity, including random residuals. As a result, its main shortcoming is that it is not applicable to nonlinear systems (Zhang *et al.*, 1998), since it does not have a mechanism to capture non-linear temporal relations hidden in the input signals. Another shortcoming is that, many steps should be performed to specify the appropriate set of parameters for Auto-ARIMA models. Despite these limitations, ARIMA models have been applied to different forecasting problems (Alghamdi *et al.*, 2019). Since our dairy dataset contains features with non-linear correlations with the target income series, the Auto-ARIMA model may struggle to accurately predict future lactation periods. This limitation arises from its inability to capture the non-linear dependencies between the input variables, target variables, and their corresponding lags. Another fundamental and classical model is the Exponential Smoothing (ETS) model (Hyndman et Athanasopoulos, 2018), that assigns exponentially decreasing weights to past observations. It works by capturing three main components of a time series, namely, level, trend, and seasonality, through additive or multiplicative formulations. ETS models are particularly effective for short-term forecasting when the time series exhibits consistent trend or seasonal patterns. Their simplicity, interpretability, and computational efficiency make them widely used in industrial applications and benchmarking studies.

3.3.2 **Deep learning and RNN models**

(Frasco *et al.*, 2020) is another univariate time-series forecasting model based on Long-Short Term Memory neural networks (Hochreiter et Schmidhuber, 1997b). The inputs to the first LSTM layer are the dairy feature vectors related to a sequence of months. For instance, the input to the first LSTM cell is the dairy factors associated with the first month of the first lactation. These factors include cow health, dairy profit, milking quantity and quality information. Two stacks of LSTM layers followed by a linear fully-connected layer are used to predict the milk production profits across the future months. LSTM captures the temporal relationships between different time-steps (in this case months) and learn the sequential data, efficiently. In addition, LSTMs are capable of learning long-term dependencies in a sequence and are less prone to the vanishing gradient problem (Bengio *et al.*, 1994), which decreases the learning capacity of a model. But, the main shortcoming of the UniMu-RNN is that only the hidden state of the last LSTM cell is fed to the linear layer in order to predict the profits and the information related to other hidden states, which are associated with the previous time-steps is neglected. In addition, this method does not utilize the spatial correlations between the variables.

3.3.3 **Deep learning and linear models**

A deep learning model (N-BEATS) is proposed by (Oreshkin *et al.*, 2019) for univariate time series forecasting. N-BEATS uses a stack of fully connected neural network layers to forecast the future values of a target series based on the past values in a single pass. It consists of several blocks connected in a residual manner. Inside each block there are two branches corresponding to backcast and forecast paths. Both forward and backward paths have several fully-connected layers followed by an activation function. Each block accepts its respective input as the residual output of the previous block except the very first block which receives the original time series as its input. Basically, the building block contains two parts : the first part is fully-connected networks (without activation function) that generate the backward and forward predictors of expansion coefficients. The second part includes the backward and forward basis layers, which project the respective expansion coefficients on the set of basis functions to produce backcast and forecast outputs. N-BEATS model consists of seasonality and trend blocks thanks to the seasonality and trend basis coefficients which give the model more power in capturing trend and seasonality of the input time series. Two architectures have been proposed including generic and interpretable architectures. The generic architecture does not depend on time series specific information and the basis functions are a linear projections of the previous layer outputs. The interpretable architecture contains both trend and seasonality blocks,

which applies trend and seasonality basis functions to the expansion coefficients in order to capture trend and seasonality, elements separately. DLinear and NLinear (Zeng *et al.*, 2023) are lightweight linear models designed for time series forecasting, proposed as efficient alternatives to deep learning-based and Transformer-based architectures. DLinear first decomposes the input time series into trend and seasonal components, then applies separate linear transformations (fully-connected layers) to each, leveraging their distinct temporal dynamics, then combining the predictions of trend and seasonality components as the final forecast. On the other hand, NLinear directly applies a linear transformation (fully-connected neural network) to the input series without decomposition. Both models demonstrate promising performance on long-term forecasting benchmarks with significantly reduced computational complexity and parameter count, making them attractive for resource-constrained applications.

3.4 Multivariate time series forecasting models

3.4.1 **Classical and statistical methods**

Vector Auto-Regression (VAR) model (Zivot et Wang, 2006) is used for multivariate time series prediction and represent each variable as a function of past values of itself and lags of the other variables. The main parameters in a VAR model are p, which means using the first p lag of all variables in a system in order to predict a specific variable and K, which represents the number of variables chosen to be used in the prediction process. VAR is a natural extension of univariate auto-regressive models for describing the dynamic behavior of multivariate economic and financial time series data. The main limitation associated with VAR models is the random walk-like behavior of VAR, when the stability condition does not hold (numerical limitations) (Kammerdiner *et al.*, 2007).

3.4.2 **Deep learning models**

The deep learning methods are generally based on Recurrent Neural Networks (RNN), linear layers, Convolutional Neural Networks (CNN), Sate Space Models (Mamba), and Transformer models. Among the RNN based models, DeepAR exploits RNN and auto-regressive decoding to predict the time series (Salinas *et al.*, 2020). SegRNN (Lin *et al.*, 2023) proposes segment-wise processing of the input sequence using RNN layers and parallel multi-step forecasting, which reduces the number of iterations to produce the forecasting horizon. Similar to the UniMu-RNN, its multivariate version (MuMu-RNN) model (Frasco *et al.*, 2020) utilizes the LSTM neural networks to predict milk production profits in the following months. MuMu-RNN consists of

two stacks of LSTM layers followed by a linear layer to predict the profits in the future months. CNN models employ convolutions to extract temporal and sub-series dependencies (Wu *et al.*, 2022) (Liu *et al.*, 2022). For example, SCINet (Liu *et al.*, 2022) utilizes multiple convolutions to extract temporal information from different down-sampled versions of the series. TimesNet (Wu *et al.*, 2022) on the other hand, converts the original one-dimensional time series into two-dimensional series using different period lengths and uses 2-dimensional convolutions to capture inter-period and intra-period information. TimesNet efficiently captures multi-resolution dependencies by using Fast Fourier Transform (FFT) of the input series and utilizing 2-dimensional convolutions for representation of inter-period and intra-period relations. Models based on Multi-Layer Perceptron (MLP) have been proposed in (Das *et al.*, 2023) and (Zeng *et al.*, 2023), which exhibit an effective performance in time series domain. For instance, D-Linear (Zeng *et al.*, 2023) decomposes a time series into trend and seasonality components and applies a fully-connected layer to directly predict the future trend and seasonality parts. Recently, very powerful linear models were designed, which utilize both temporal and cross-channel dependencies. TSMixer (Chen *et al.*, 2023a) employs interleaving temporal and feature mixing MLPs to represent temporal and inter-channel correlations. More recently, TimeMixer (Wang *et al.*, 2024b) was proposed based on the idea that time series exhibit distinct patterns in various sampling scales. They employ an MLP-based multiscale mixing model, which aggregates trend and seasonality components of different scales hierarchically and leverages multiscale information in an ensemble fashion to deliver the predictions. Several works have utilized Graph Neural Networks (GNN) for time series forecasting, specifically for traffic forecasting. For example, MTGNN (Wu *et al.*, 2020b) utilizes temporal and graph convolutional layers to capture temporal and spatial (cross-channel) dependencies. An attention based spatial-temporal graph neural network is proposed in (Guo *et al.*, 2021) which captures the temporal dynamics of traffic series by using the local context. In (Wang *et al.*, 2022), a feature correlation-aware spatio-temporal graph convolutional network is designated for traffic prediction, which captures multi-scale spatial and temporal relations effectively, considering cross-scale dependencies. Dynamic graph structure learning for multivariate time series forecasting (Li *et al.*, 2023b) exploits graph learning networks to capture hidden dependencies between variables, enhancing the accuracy of forecasting by effectively capturing complex interrelationships within the data. Another work (Zhao *et al.*, 2023), addresses the problem of capturing dynamic correlations by learning historical relation graphs and predicting future relation graphs. They also designed a causal GNN for feature extraction and reasoning network to capture the relations between historical time steps and forecasting horizon.

With their outstanding breakthrough in Natural Language Processing (NLP) and Computer Vision (CV) fields,

Transformer models have recently shown superior performance in time series forecasting task and they have been continuously evolving. Among them, the Informer model (Zhou *et al.*, 2021) is designed for long sequence time-series forecasting to address issues related to the Transformer model, such as quadratic time complexity and high memory usage. First of all, Prob Sparse Self-Attention is proposed in Informer to alleviate memory and time-complexity issues related to the canonical self-attention module used in Transformers. Prob Sparse Self-Attention allows each key to only attend to most prominent queries. Queries with larger Kullback-Leiber divergence between the query-key attention probability distribution and the uniform distribution are selected as the dominant queries. The second contribution of Informer model is self-attention distilling operation which are used to connect the self-attention blocks and down-sample the output of the previous self-attention block into half-size by using convolutional and max-pooling layers. Third, Informer could contain extra encoders by progressively decreasing input length and number of self-attention modules. More than above, Informer model exploits a generative style decoder capable of predicting outputs in one forward pass. In Autoformer (Wu *et al.*, 2021), the self-attention is replaced with auto-correlation to capture temporal dynamics by utilizing Fast Fourier Transform. FEDformer (Zhou *et al.*, 2022) deploys Fourier transformation to deal with time series data given the fact that time series tend to have a sparse representation in Fourier basis. Recently, the PatchTST model (Nie *et al.*, 2022) was proposed based on patching and channel independence to model time series data, pinpointing that Transformer architecture is still a powerful model with some adaptation and architectural adjustments. Following PatchTST, other Transformer models have been developed for time series forecasting and proved high capability in dealing with high-dimensional time series (Liu *et al.*, 2023). iTransformer (Liu *et al.*, 2023) utilized multi-head self-attention to capture complex cross-channel dependencies among time series channels for the first time and delivered promising results, highlighting the capability of self-attention in channel-wise modeling of time series. Multi-scale representation proved to be necessary for time series analysis (Cirstea *et al.*, 2022) (Shabani *et al.*, 2022). Triformer (Cirstea *et al.*, 2022) suggests a patch attention with linear complexity and variable specific parameters to enhance accuracy. (Shabani *et al.*, 2022) develops a multi-scale framework to model time series using different resolutions and they utilize separate predictive models for each temporal scale which leads to high computational complexity.

After the rise of generative pre-trained models, Large Language Models (LLMs) have been utilized for time series forecasting by fine-tuning and exploiting prompt engineering, e.g., Time-LLM (Jin *et al.*, 2023) which keeps the backbone LLM intact and input time series is reprogrammed with text prototypes before feeding it to the frozen LLM, aligning two modalities. GPT4TS (Zhou *et al.*, 2023) employs GPT-2 (Radford *et al.*, 2019)

model for time series forecasting by feeding the time series patches to the model in a channel-independent manner. A spatial-temporal large language model is proposed for traffic forecasting (Liu *et al.*, 2024) by defining spatial-temporal embedding to learn the spatial locations and global temporal dependencies of time steps at each location. In traffic prediction often capturing spatial-temporal dependencies at multiple scales is required. To address the mentioned requirement, MT-STNets is designed in (Wang *et al.*, 2021b), for prediction of both fine-grained traffic conditions on individual roads and coarse-grained traffic flows across urban areas.

More recently, Pathformer (Chen *et al.*, 2024) is proposed which exploits adaptive pathways to capture multi-scale temporal relations in an adaptive manner by automatically selecting patches of different resolutions and applying the attention blocks to capture inter-patch and intra-patch dependencies. This model uses separate set of parameters for each temporal granularity in its architecture. As mentioned earlier, iTransformer (Liu *et al.*, 2023) model is proposed based on the vanilla Transformer model and embedding the channels of the input series along the temporal dimension. This model captures cross-channel dependencies through the multi-head self-attention blocks and temporal correlation by using feed-forward layers, pinpointing the importance of modeling cross-channel interactions in the forecasting tasks.

State Space Models (SSMs) have recently garnered significant interest from researchers because of their robust sequential modeling capabilities and interpretability. Among these, the Mamba model (Gu et Dao, 2023) stands out, utilizing a hierarchical Bayesian framework to effectively capture complex temporal correlations and adapt to evolving dynamics over time (Gu et Dao, 2023). The more recent advancements have seen the Mamba model applied to capture cross-channel dependencies in time series data, yielding promising results in time series forecasting (Wang *et al.*, 2025). More recently, a general pattern machine is proposed (Wang *et al.*, 2024a) which utilizes multiple scales in the time domain and different resolutions in the frequency domain, employing 2-dimensional convolutional mixing strategies for extraction of intricate and task-adaptive time series patterns. TimeMixer++ relies on column-wise and row-wise attention modules which act on the time images of various resolutions and scales to extract cross-period and inter-frequency dependencies across various scales, mixing the scales through 2-dimensional convolution and deconvolution operators in bottom-up and top-down hierarchies.

3.5 Dairy prediction use cases

Prediction of milk yield helps farmers to expand their forecasting horizon with respect to their farm's future profitability. Therefore, forecasting milk yield is a prominent asset for farmers which can lead to optimal decision making in herd management (Dematawewa *et al.*, 2007). For instance, lactation models lead to predict income associated with dairy farms more efficiently as shown in (Ehrlich, 2011).

3.5.1 Calving prediction from activity, lying, and ruminating behaviors in dairy cattle

Calving time is predicted based on automated activity, lying and rumination monitors corresponding to Holstein dairy cattle (Borchers *et al.*, 2017). They used automatically collected neck activity and rumination data in two hour intervals (increments) together with automatically measured number of steps, lying time, standing time, number of transitions from standing to lying and total motion. All of these behavioral data were collected for 14 days before the predicted calving dates. Mixed linear models were used to perform retrospective data analysis and behavioral changes. They also evaluated bi-hourly behavioral differences from baseline values over the 14 days before calving. They concluded that extreme values for all behaviors occurred in a 14-day period before calving. Different machine learning methods, such as neural network, random forest, linear discriminant analysis are selected and trained using the behavioral variables. These models are used to determine two hour periods in 8 hours before calving time.

3.5.2 Quantifying shape of lactation curves for common dairy breeds and parities

(Ehrlich, 2011) proposed a lactation model for both the shape and magnitude of the lactation curve as a set of parameter values that describe the shape of the lactation curve. Lactation data are used to train the model to summarize a lactation in terms of parameter values. The scale parameter describes the magnitude without having any impact on the shape of the curve. The ramp parameter controls the steepness of the rise in milk production. The decay parameter is responsible for controlling the rate of late lactation decline, and the offset parameter specifies a theoretical offset value between the beginning of milking period and calving. In this model, offset is described in terms of time (day) and defines the time when the productive capacity is maximally created.

3.5.3 A comparison of neural network and multiple regression predictions for 305-day lactation yield

In (Grzesiak *et al.*, 2003) ANN and multiple regression models are used to predict 305-day lactation yield based on partial lactation recordsan the prediction results from both models are compared. The ANN model has 7 neurons in the input layer as they utilize seven dairy variables. These variables or features include, average 305-day milk production of the barns, days in milk, average test day milk yield in the first, second, 3rd and 4th sampling months (4 factors) and month of calving which was encoded. The ANN model had 10 neurons in the hidden layer and 1 neuron in the output layer. The use of neural networks was suggested in (Grzesiak *et al.*, 2003) as an alternative method to predict dairy traits.

3.5.4 Leveraging latent representations for milk yield prediction

Recently, Liseune et al. (Liseune *et al.*, 2020) proposed a deep learning model to predict the dairy milk yield curve. In their approach, the lactation curve is reconstructed by forecasting and decoding a low-dimensional representation derived from a convolutional neural network (CNN)-based feature extraction process. The input time series, composed of milk yields and embedded production and health events, is first passed through a sequence of convolutional blocks. Each block consists of a convolutional layer followed by a non-linear activation function, which slides over the sequence to extract salient temporal features. Pooling layers are subsequently applied to reduce dimensionality and retain the most informative aspects of the sequence.

Before being input to the CNN, categorical events are embedded using a trainable embedding layer. Max pooling is used on the hidden representations to extract the most relevant features within specific time intervals. The output from the final convolutional layer is flattened to form a static representation of the event sequence, which is then passed to an autoencoder (AE), which learns to encode and decode the most informative traits of the lactation curves. The AE output retains the same dimensionality as the original CNN feature vector. To reconstruct the lactation curve from the abstract feature representation, a deconvolutional neural network (DNN) is employed. The DNN acts as the inverse of the CNN, mapping the static features back into a one-dimensional sequence of predicted milk yields. This architecture, referred to as a sequential autoencoder (SAE), integrates CNN, AE, and DNN components to model and impute missing milk yield values. To evaluate performance, a multilayer perceptron (MLP) baseline was also used for comparison. The SAE improves the results of the MLP in terms of the coefficient of determination (R^2), indicating its superior ability to reconstruct and forecast milk yield trajectories.

3.5.5 Estrus detection system of dairy cows

Monitoring estrus of dairy cow accurately is a key factor in improving dairy farms (Ma *et al.*, 2020). In order to realize cost-effective detection systems, Narrow-band Internet-of-Things (NB-IoT) is accounted as a promising technology. In (Ma *et al.*, 2020) a NB-IoT based framework has been proposed to predict estrus state of dairy cows by analyzing multiple time series related to behaviour characteristics of dairy cows, such as the number of steps, minutes of dairy cows in high activity, medium activity, low activity, feeding, and rumination. The deep learning architecture proposed for estrus prediction is composed of CNN layers and LSTM networks. A simple convolutional network is utilized to capture the local dependencies among the dairy variables (characteristics). Then, the extracted features of the convolutional layer are fed into the recurrent component (LSTM layer) to obtain the temporal relations between the series at different time-steps. Finally, the recurrent component is coupled with a dense layer to provide the final prediction. The estrus detection network was used to predict estrus state in the next step based on the prefix sub-sequences of length k. This model is not applicable to multi-horizon forecasting problems, because it relies on a sub-sequence of fixed length to predict the estrus state in one step ahead. As a result, this method is not suitable for applying to our profit forecasting task during several months ahead. Another shortcoming is the lack of using all the hidden states to capture temporal correlations between all the input time-steps.

According to the presented literature review, machine learning and deep learning are highly applicable in the dairy field and have led to promising results in calving prediction, milk yield forecasting, lactation curve estimation, and estrus detection.

Considering the advantages and shortcomings of these methods, I will design a model that addresses their limitations while leveraging their strengths. Most of those models focus on the temporal dependencies in the input series, without considering the cross-channel correlations between the input variables. However, I find it is important to consider these interactions among the variables or multiple series to predict the future values of a target series. On the other hand, sequence based models demand a high time complexity, especially when dealing with very long sequences. Transformer-based models capture the long-term temporal dependencies in an efficient manner, but time complexity of the self-attention blocks inside the models are quadratic in sequence length. Considering the great capabilities of the Transformer model, I will add a cross-channel encoder module to it for capturing the relations between multiple series and simultaneously representing the long-term temporal dependencies. Based on the reviewed models, time complexity of the Transformer model could be improved by estimating the attention matrix or selecting the most relevant

queries in the query-key evaluation part inside the self-attention paradigm. In addition, most real-world time series exhibit multiple resolutions and are sampled at various scales (Chen *et al.*, 2024). Therefore, incorporating multi-scale information in time series modeling leads to a more comprehensive analysis and improves forecasting accuracy.

In the following chapters, I will describe the approaches that I proposed in order to address the objectives obj1, obj2 and obj3 in more detail.

CHAPITRE 4

UNIVARIATE AND MULTIVARIATE TIME-SERIES METHODS TO FORECAST DAIRY INCOME

In this chapter, several models for time series forecasting of dairy production are presented, including a novel model based on LSTM and attention mechanisms. These models are introduced to address Hypotheses 1 through 5. The performance of various models is assessed in terms of error measures and statistical tests to verify the efficacy of the deep learning models in the prediction of future milk production and to show their superior performance compared with the statistical models. In addition, herd-wise errors are reported for each model to assess their performance at herd level, and their prediction errors are compared when predicting milk income in one month ahead. In addition, we demonstrate the running time and the parameter count for each model to evaluate their time and memory complexity. Standard recurrent models summarize long sequences through their final hidden state, which can dilute biologically important events occurring at earlier stages of the lactation cycle. The intuition behind MuMu+Attention is that not all time steps contribute equally to profitability prediction. By introducing an attention layer over the hidden states of stacked LSTMs, the model explicitly emphasizes informative periods while down-weighting less relevant time-steps. This allows the model to focus on key physiological and management events that influence dairy income, improving robustness compared to conventional RNN-based approaches. The study presented in this chapter was published in the 2nd AAAI Workshop on AI for Agriculture and Food Systems proceedings under the title "Univariate and multivariate time-series methods to forecast dairy income". Article writing, approach design, implementation, and experimental design were performed by Vahid Naghashi, under the supervision of professors Abdoulaye Baniré Diallo and Mounir Boukadoum. A printed version of this paper is represented in Appendix A.

RÉSUMÉ

Forecasting the income from milk sales can be addressed as a time series problem since the sequence of multiple dairy attributes during lactation cycles are inter-related and temporally dependent. In this paper, we provide a framework to forecast the income from milk sales during the third lactation of the dairy cows based on dairy attributes recorded through the first and second lactation. We modeled the problem as univariate and multivariate time series predictions. We propose several state-of-the-art implementations with ARIMA, N-BEATS, transformer and an original method, MuMu+attention, that combines Long-Short Term Memory neural network and attention mechanism to capture the temporal dependencies. To benchmark the implemented methods, we curated data from 147,749 dairy cows from 5,844 Canadian herds. The monthly income from milk sales ($CAD) measured at each cow during their third lactation was treated as the prediction target. The dataset was composed of dairy attributes of milk quality, production, season, year, and health recorded over the first and second lactation of the dairy cows. The results highlight that most of the methods can achieve relative good performance with the best prediction accuracy obtained by MuMu+attention. MuMu+attention results are 43% better over the classic ARIMA model. By forecasting the income from milk sales, our model could help farmers to early identify less profitable animals and better allocate resources.

4.1 Introduction

Dairy farming is one of the largest sectors in agriculture that has been under study through data-driven and machine learning methodologies. Promising results were obtained in automatic cropping of the cow's body region and cow's pattern identification for individual animals (Zin *et al.*, 2018). In another work, deep convolutional neural networks were used for detection of the key parts of a dairy cow's body, resulting in an accurate detector (Jiang *et al.*, 2019). Promising results were obtained by exploiting a deep learning model for calving prediction from activity, lying, and ruminating behaviors of dairy cattle (Borchers *et al.*, 2017). Therefore, the use of machine learning and deep learning based models lead to promising results and its application to predict dairy production could yield satisfying results (Frasco *et al.*, 2020).

Income from milk sale is the main factor associated with the profitability of a dairy farm. A forecasting tool would allow farmers to optimize the allocation of resources by early identifying and removing less profitable animals from the herd. The problem of forecasting income from milk sales can be modeled as a univariate time-series task since the lactation cycles are inter-related and temporally dependent, or multivariate since milk production, the consequently income, is affected by heath, productivity, environmental, and management conditions. Among the classical time-series prediction methods, Autoregressive Integrated Moving Average (ARIMA) has shown a good performance in univariate time-series prediction tasks (Contreras *et al.*, 2003). Recently, deep learning models have been exploited in time-series prediction domain. N-BEATS (Oreshkin *et al.*, 2019) is one of the well-known deep prediction models, in which residual connections are used

for univariate time-series prediction and the model's architecture is based on a very deep stack of fully-connected layers. In MuMu (Frasco *et al.*, 2020), Long-Short Term Memory (LSTM) network was exploited in the dairy forecasting field and gave rise to auspicious results.Recently, Transformer models (Vaswani *et al.*, 2017) got more attention from researchers due to their ability to represent the long-term temporal dependencies efficiently by incorporating multi-head self-attention in their structure. The efficacy of the self-attention layers gave us the hint about using an attention module in our prediction framework.

The objective of this paper is twofold : first, we propose an extension of MuMu using LSTM and an attention mechanism to forecast the income from milk sales ; second, we benchmark univariate models against multivariate ones in predicting future profit using a well curated data from 147,749 dairy cows.

4.2 Preliminary

The problem of lifetime milk revenue prediction can be posed as follows. For each input example (ith cow sample) of length T, i.e. $x_i = (x_i^1, ..., x_i^T) \; \epsilon \; \Re^{p \times T}$ with p as the number of input dairy factors and T the length of the time series (total length of the first and second lactation fixed for all the cow samples), a prediction model forecasts the upcoming milk revenues of M steps (months) ahead in third lactation, $\hat{r_i} \; \epsilon \; \Re^M$. Therefore, the goal is to learn a function f which maps the input multivariate time-series $X \; \epsilon \; \Re^{s \times T \times p}$ to the estimated milk income values in the future lactation months $\hat{r} \; \epsilon \; \Re^{s \times M}$, with s being the number of the cow samples presented to the model as the training set : $\hat{r} = f(X)$.

4.3 Methodology

Prediction of dairy income can be stated as four different cases (Figure 4.1) : univariate or multivariate inputs, single or multiple outputs. In the first case, the input is a sequence of input dairy incomes during the early lactation cycles and the output is the value of the next income in the next lactation. The second case is similar to the first one, with the difference that the output is multiple dairy incomes. In the third case, the input window consists of multivariate series associated with the multiple dairy factors through the early lactation and the output window corresponds to a single income in the next lactation. Finally, the input window in the fourth case contains multivariate series of multiple dairy attributes similar to the latter case, and have a sequence of dairy incomes as its outputs.

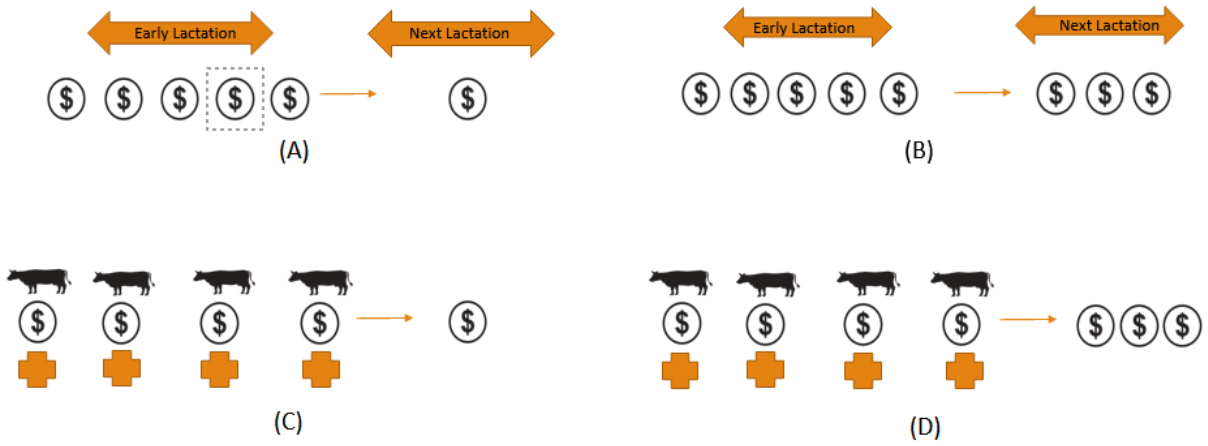


Figure 4.1 – Univariate and multivariate statement of our dairy prediction problem.

4.3.1 Model architecture

Here we propose MuMu+Attention, which builds on top of the MuMu model (Frasco *et al.*, 2020). It implements time series of the individual dairy attributes corresponding to earlier lactations. It consists of two LSTM layers stacked on top of each other and one attention layer, followed by a linear layer acting as a decoder. The architecture of the proposed framework is illustrated in Figure 8.1.

For the purpose of using all the hidden states and temporal feature selection, an attention layer (Vaswani *et al.*, 2017) has been embedded on top of the last LSTM layer in our model. The attention layer consisted of two linear layers and one $tanh$ activation function in between. The final attention weights were the result of applying a $softmax$ function to the output of the second linear layer to normalize the attention scores. These weights were multiplied by the corresponding hidden states and a weighted hidden vector was calculated in order to be fed into the final linear layer which generates the final prediction over the target window.

The corresponding formulas describe the components of the attention layer :

$$L_1 = tanh(W_1 x + b_1) \tag{4.1}$$

$$Attention\_Outputs = Softmax(W_2 L_1 + b_2) \tag{4.2}$$

In the above formulas, x represents the output of the last LSTM layer with the shape of (batch_size, sequence_length, hidden_size). $W1$, $W2$, $b1$ and $b2$ are learnable parameters which have been trained in an end to end manner together with the other parameters of the model.

The major advantage of using the attention layer is that the information in all of the hidden states has been exploited, rather than using only the output of the last LSTM cell and this can be referred to as a temporal feature selection step which learns and assigns the importance weights to the sequence of the hidden states associated with the input window (input time steps). A dropout layer is used in the output of the second LSTM layer in order to avoid over-fitting. The value of the dropout rate hyper-parameter determines the ratio of the hidden states whose outputs are dropped out and this hyper-parameter is set to 0.5 in our model. Finally, the decoder in our model, which is a linear layer with no activation function, generates the output predictions by one forward pass instead of the time consuming dynamic decoding used in the conventional architectures. In our work, the mean squared error was used as the loss function while training the model.

4.4 Experimental settings

4.4.1 Data

The input to our prediction model was a multivariate time series containing a set of dairy attributes, including metrics of milk quality, seasonality, year, health, and management factors, recorded during the first and second lactation of 147,749 dairy cows from 5,844 Canadian dairy farms over the years of 2006 to 2017. The prediction targets were the monthly income from milk sales ($CAD) measured at each cow during their third lactation.

4.4.2 Data pre-processing

In the experiments, lactation length was fixed to 11 months for first, second, and third lactation based on the mean + one standard deviation of the lactation lengths related to the training cow samples. For cows with lactation lengths shorter than 11 months, additional data rows were created in the missing months and imputed through linear interpolation based on the two closest months. The following steps were taken to clean the data : 1- Keeping only animals having test records in the first, second and third lactations, 2- Deleting the records from the dry period : dry period is defined as the months in which a cow doesn't produce milk (which mostly occurs between two lactation cycles) and milk value (income) is almost zero. Since

there is no income from sales during the dry period, records associated with these months were deleted, 3- Removing the animals who left their herd before and during the third lactation, 4- Deleting duplicate records, 5- Deleting the records with negative milk value and cumulative milk value : The reason for removing such records, which constituted a small percentage of the data, was to remove the inconsistencies in the data set, as the milk value is the income from milk sale (CAD) and negative values of dairy income could be accounted for as an inconsistency due to the errors in data acquisition steps, 6- Deleting the records which included contradictions, e.g., rows indicating the cow was milking, while the milk yield of those records was equal to zero, 7- Outlier removal (their corresponding rows) : The values outside the range of Mean -/+ 2.5 $\times$ standard deviation were specified as outliers for a given dairy attribute and were deleted. All the dairy attributes used as input to our model and their descriptions are shown in Tables 4.3 and 4.4 (Appendix).

Among the 147,749 dairy cows, 100,000 were selected to train and the remaining 47,749 cow samples to test the models. Missing data was imputed after train-test split to avoid information leakage (Thomas *et al.*, 2020). In our work, missing value imputation were performed in 7 consecutive steps : first the cow samples were grouped based on their herd ID, season and year, then the missing values within each group were imputed using the average of non-missing samples within the same group. The imputation process in the remaining six steps was similar as in the first one, with the following attributes used for grouping, respectively : (herd ID, season), (herd id, year), (herd id), (season, year), (season), (herd id).

4.4.3 Experimental details

4.4.3.1 Baselines

We choose five forecasting methods as comparison models : two univariate (ARIMA (Contreras *et al.*, 2003) and N-BEATS (Oreshkin *et al.*, 2019)) and two multivariate models (MuMu (Frasco *et al.*, 2020) and a standard Transformer model (Vaswani *et al.*, 2017) adapted for time series prediction. Most of the above methods are well known in the time series domain. The ARIMA model is chosen as a baseline method as it is a classic statistical approach in time series analysis. The parameters used in the ARIMA are taken from (Frasco *et al.*, 2020) which used a stepwise algorithm to determine p, q, P and Q parameters. The N-BEATS model (Oreshkin *et al.*, 2019) had four blocks. The first two blocks had a trend basis function in their outputs, and the other two contained a seasonality basis. The number of fully-connected layers inside each block was fixed to 4 with hidden-size of 128. As the N-BEATS model is designed to receive a univariate sequence, we fed the sequence of milk incomes in 22 months as its input. The parameters for the MuMu model are : 2

LSTM layers, one linear layer (without an activation function) and the hidden size is fixed to 32 in all layers. The Transformer model which was used as another baseline method in our experiments, was composed of two encoders and one decoder layer, in which the input of the decoder was the last time step of the input window.

4.4.3.2 Grid search for hyper-parameters determination

Grid search was conducted to determine the hyper parameters (batch size, learning rate and hidden size of the LSTMs) due to its common usage in other related works and satisfying results (Zhou *et al.*, 2021). Our proposed model was optimized with Adam optimizer and learning rate of $1e^{-4}$. The total number of epochs was set to 20 with an early stopping based on the validation loss. **Setup :** All the numerical inputs were standardized with zero mean and unit standard deviation. At the same time, categorical variables were normalized to the range between 0 and 1. The above transformation was applied to each dairy feature, separately. The input window was set to 22 (with the length of 11 for both of the first and second lactation). The prediction (target) window size (length of the third lactation) was fixed as 11 in our experiments according to the mean of the lactation lengths corresponding to all the cow samples. We evaluated our prediction framework using the $RMSE = \sqrt{\frac{1}{N_{cows} \times N_{months}} \sum_{m \epsilon months} \sum_{c \epsilon cows} (\hat{p}_{c,m} - p_{c,m})^2}$ and the $MAE = \frac{1}{N_{cows} \times N_{months}} \sum_{m \epsilon month} \sum_{c \epsilon cows} |\hat{p}_{c,m} - p_{c,m}|$ on the forecasting window. All the models were trained and tested on 4 NVIDIA T4 Turing GPUs with 16 GB GDDR6 memory. Additionally, multiple pairwise Wilcoxon tests, with Bonferroni p-value adjustment, were used to compare models.

4.5 Results and discussion

Tables 4.1 and 4.2, summarize the forecasting accuracy of all the methods on the test dataset. The best results are highlighted in boldface in each setting. We also reported the chosen hyper-parameters (the best combination), including hidden size, learning rate and batch size after performing the grid search in Table 4.5 (Appendix).

MuMu+Attention model was able to forecast the income with the highest accuracy based on the RMSE (Table 4.1) and MAE (Table 4.2) measures. The LSTM layers included were able to represent and capture the long-term temporal dependencies (Gers *et al.*, 2000) by exploiting the input, forget, update, and output gates in its structure. Those gates help the LSTM to select the most relevant information and update the previous state using the current input at each time step. At the same time, forget and update elements give

Table 4.1 – Prediction results in terms of RMSE (test dataset).

RMSE results				
Univariate		Multivariate		
ARIMA	N-BEATS	Transformer	MuMu	MuMu+Attention
7.094	4.198	4.064	4.082	**4.052**

Table 4.2 – Prediction results in terms of MAE (test dataset).

MAE results				
Univariate		Multivariate		
ARIMA	N-BEATS	Transformer	MuMu	MuMu+Attention
5.560	3.256	3.178	3.175	**3.143**

the LSTM the capability of remembering the long-range dependencies and making use of such relationships in the prediction process.

Among the baseline methods, the classical ARIMA forecasts had the highest error. The linear nature of this model hinders its ability to capture more complex and non-linear temporal correlations. The N-BEATS model had a relatively higher error compared to the multivariate approaches, which is likely because it does not have a sequential module in its structure to represent the temporal dependencies. Capturing long-range dependencies is a key factor in time-series prediction (Zhou *et al.*, 2021) and all the LSTM or Transformer based approaches try to represent such relations by capturing the impact of different time steps on each other, which seems to be a dominant factor for the success of those methods. The reason behind the relatively lower performance of the Transformer model might be due to the lack of sufficient and enough data for training of this model, which has more parameters than the other models (Table 4.9). As the errors related to different deep learning models are in the same range, the training and inference time of the MuMu+Atten and other models are reported in Table 4.6 to give an insight on the time complexity of the deep learning based methods. Based on the results, the N-BEATS model needs more training and inference time than the MuMu+Atten, despite its lower performance compared to the multivariate models. We further conducted other experiments to predict the milk value in a single month after the first and second lactation (input window). The results indicate that the model nearly gave the same results as the prediction of the multiple months (Table 4.7).

Figure 4.2 depicts the forecast accuracy of the proposed framework in comparison to the other models over each month of the third lactation. Except for the ARIMA model, the best forecasts occurred in the

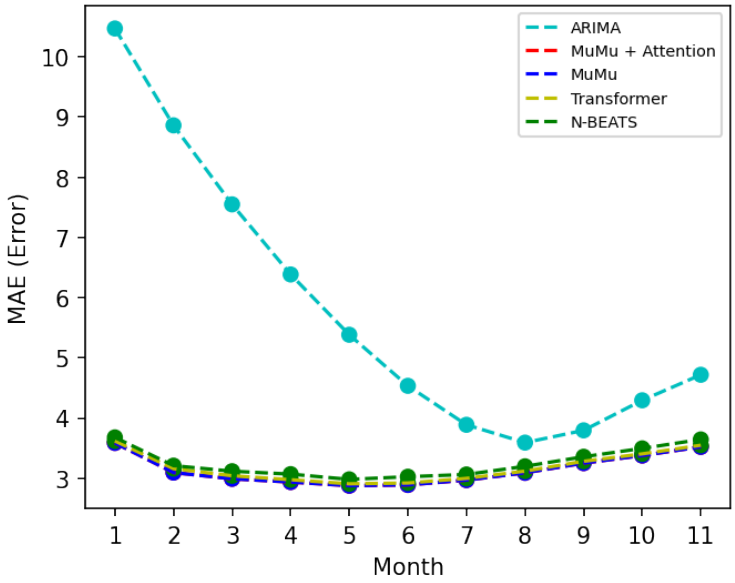


Figure 4.2 – Monthly Errors (MAE) of different methods and our proposed framework on prediction of the milk incomes.

middle of lactation (month 5) compared to the beginning and the end. This is likely because there is a great variability among cows in the amount of milk produced and consequently sold during these steps, making it more difficult to forecast. The distribution of MAE for all models at the herd level is plotted in Figure 4.3. There was no strong evidence of herd-bias as the distributions were visually similar. The MAE, at the herd level, was not statistically different between the multivariate models ($p \geq 0.05$) and it was lower than both univariate models ($p < 0.05$; Table 4.8). This indicated that the proposed framework could also capture the long-range temporal dependencies in input and output windows by processing the input sequence using the LSTM layers and representing the importance of each time step through the attention weights. Furthermore, the generative style decoder (the output linear layer in our framework) acquired the output predictions in one forward pass and avoided the error accumulation during the testing phase.

4.6 Conclusion

In conclusion, we studied the problem of forecasting income from milk sales by designing a framework and comparing univariate and multivariate approaches. We enriched our framework with the MuMu+Attention model that combines LSTM and attention mechanism. The experimental results showed that multivariate models tend to perform better even though the performance of NBEAST can be an important way of approxi-

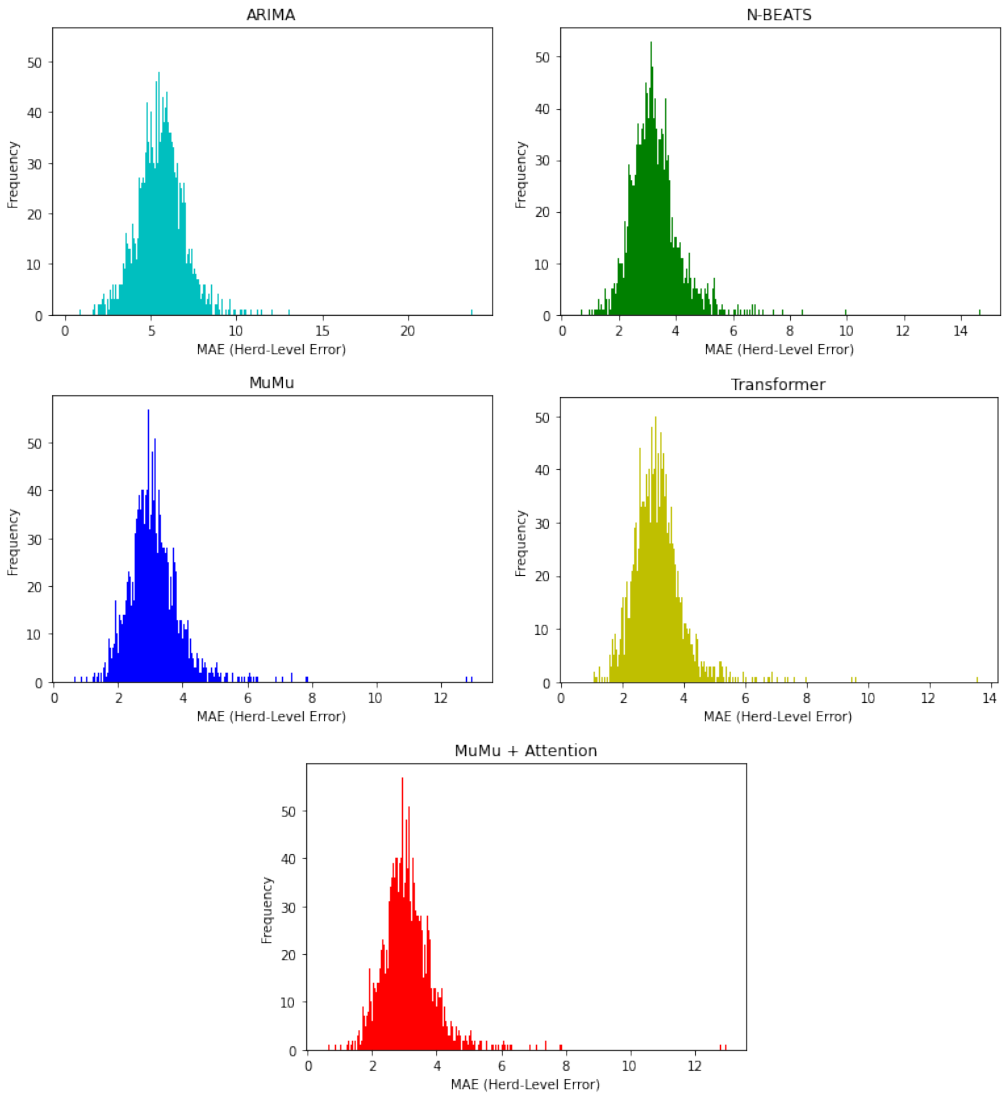


Figure 4.3 – Herd-based Error (MAE) distribution of different methods.

mating the profit just using a univariate time series. However, we showed that MuMu+Attention provided the highest accuracy.

Appendix

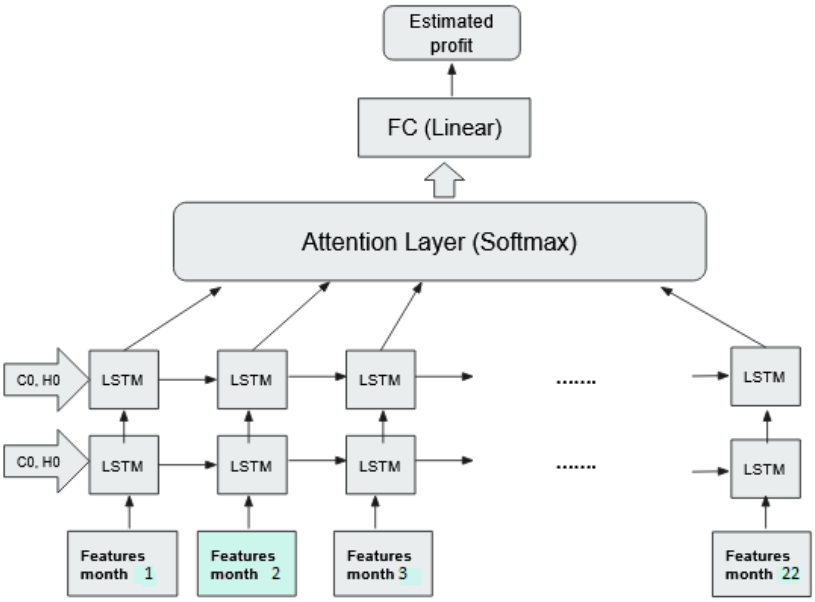


Figure 4.4 – Architecture of the proposed prediction framework.

Table 4.3 – Numerical dairy variables used in our framework.

Variable	Missing %	Min	Max	Mean	Std
Milk produced in 24 hours (Kg)	1.4	0.20	61.40	29.60	9.25
Milk income (CAD)	0.002	0.09	42.95	21.21	5.94
Somatic Cell Count (1000/mL)	17.62	1	1754	156.59	254.87
Milk Urea Nitrogen	56.04	2.10	19.80	10.91	3.27
Lactose yield (Kg)	37.63	0	6.30	4.55	0.22
Fat yield in 24 hours (Kg)	1.64	0	2.40	1.19	0.38
Protein yield in 24 hours (Kg)	1.64	0	1.94	1.00	0.30
Number of days cow has been in milking	0	0	517	169.56	99.39
Fat to protein ratio	1.64	0.73	1.67	1.20	0.16

Table 4.4 – Categorical dairy variables used in our framework.

Variable	Missing (%)	Level	Observations	% of Level
Lactation Number	0	1	1625239	33.33
		2	1625239	33.33
		3	1625239	33.33
Number of Milking per Day	0	1	163201	3.33
		2	4650067	95.37
		3	62449	1.28
Test Season	0	1	1235340	25.33
		2	1226102	25.14
		3	1220232	25.03
		4	1194043	24.49
Birth Season	0	1	1109460	22.75
		2	1325577	27.18
		3	1274922	26.14
		4	1165758	23.91
Animal Condition	0	1	4846703	99.40
		4	29014	0.06
Test Year	0	6	155074	3.18
		7	304329	6.24
		8	443724	9.10
		9	486607	9.98
		10	472280	9.68
		11	479705	9.83
		12	498897	10.23
		13	511067	10.48
		14	521706	10.70
		15	486234	9.97
		16	331883	6.81
		17	170715	3.50

Table 4.5 – The selected hyper-parameters in training of our proposed model.

Hyper-parameter	Selected value
Hidden size	32
Learning rate	$1e^{-4}$
Batch size	1
Epochs	20

Table 4.6 – Running time of different models (Training + Inference time). In this table, L is the length of the time series, d in the model's dimensionality, c is a multiplier, and p is the order of the ARIMA model.

ARIMA	N-BEATS	Transformer	MuMu	MuMu+Atten
$>$ 12 hours	4 hours, 7 mins	2 hours, 34 mins	1 hour, 36 mins	2 hours
$O(L * p^3)$	$O(c * L * d)$	$O(L^2 * d)$	$O(L * d^2)$	$O(L * d^2)$

Table 4.7 – RMSE and MAE of the forecasting the milk income (value) in a single month ahead of the input window (first and second lactation)

	One month RMSE	One month MAE
N-BEATS	4.72	3.72
Transformer	4.58	3.60
MuMu	4.57	3.62
MuMu+Attention	4.56	3.61
ARIMA	13.53	11.81

Table 4.8 – Pairwise comparisons of the mean absolute errors for all models at the herd level using the Wilcoxon rank sum test. The Bonferroni method was used to adjust the p-values for multiple comparisons.

	ARIMA	N-BEATS	Transformer	MuMu
N-BEATS	$<$ 0.001			
Transformer	$<$ 0.001	$<$ 0.001		
MuMu	$<$ 0.001	$<$ 0.001	0.05	
MuMu+Attention	$<$ 0.001	$<$ 0.001	0.05	1.00

Table 4.9 – Number of parameters of different models.

Model Name	Number of parameters
N-BEATS	109325
Transformer	276587
MuMu	15339
MuMu+Attention	16428
ARIMA	3
Crossformer	1064328
Transformer-BEATS	335279

CHAPITRE 5

TRANSFORMER-BEATS : A TRANSFORMER MODEL FOR TIME SERIES PREDICTION OF DAIRY MILK PRODUCTION

In this chapter, a time series forecasting model is demonstrated for time series prediction of daily milk income across the future lactation period. The model captures temporal correlations and cross-channel dependencies by utilizing self-attention blocks inside the temporal and spatial encoders, then aggregating them to predict future milk income. The proposed model (Transformer-BEATS), stacks encoder layers in a residual manner to propagate useful portion of the signal into the next Transformer block. The efficiency of the model is evaluated in terms of different error metrics and training time. Transformer-BEATS outperforms most state-of-the-art models across various forecasting scenarios. To evaluate its efficiency and generalizability, we applied the model to forecast multiple time series datasets. This work addresses Hypotheses 1, 2, 3, and 4 by handling the missing values and proposing a model tailored to forecast future milk production. Recurrent architectures process sequences sequentially, which limits their ability to capture long-range dependencies for long histories. The core intuition of Transformer-BEATS is that temporal dependencies and inter-variable relationships can be modeled more effectively through self-attention operating in parallel. By leveraging Transformer encoders, the model directly relates distant observations and variables without relying on recurrence. This design improves scalability, enhances the representation of long-term production patterns, and captures inter-variable dependencies compared to LSTM-based models. The study represented in this chapter was presented at the ISCB-Latin America SoIBio CCBCOL Conference on Bioinformatics 2024 under the title "Transformer-BEATS : A Transformer model for time series prediction of dairy milk production". Article writing, conceptualization, model design, implementation, experiments, and presentation were performed by Vahid Naghashi, under the supervision of professors Abdoulaye Baniré Diallo and Mounir Boukadoum. A printed version of this paper (poster) is provided in Appendix B.

RÉSUMÉ

Predicting the milk production of dairy cows is important for precision livestock management. It can be achieved by observing the historical volume of each cow and features encoding its health status and the farm environment. This problem can be modeled as a multivariate time series forecasting task, which demands capturing temporal dependencies and inter-series relations effectively. The current time series models for dairy forecasting often fall short to capture these intricate temporal dependencies and inter-series (spatial) relationships. Here, we present a Transformer-based architecture that effectively captures these spatial and temporal dependencies through self-attention components, with a new adaptive way to integrate the forecasts from the temporal and spatial encoders. The proposed model is enhanced by connecting two Transformer encoder blocks in a backcast (residual) manner, ensuring that the most relevant information is provided to the following block. Our approach yields better results, particularly in terms of average prediction error per cow, than the state-of-the-art methods when predicting milk production in upcoming lactation of 47,000 cows.

5.1 Introduction

Over the past decades, global food consumption has seen a significant surge, creating a pressing need for farmers to enhance their production efficiency (Bruinsma, 2017). Among the key agricultural sectors, dairy farming stands out with its substantial contribution to milk production. Recognizing the vital role of the dairy industry in both human life and agriculture, it is of paramount importance to equip farmers with the latest technological tools to help achieve their goals (Frasco *et al.*, 2020). These tools also reduce costs and increase the overall production efficiency, not only for the farmers' benefit, but also to ensure a reliable and sustainable supply of dairy products to meet the growing demand by the world population. Deep learning methods provide reliable predictions about calving, disease and productivity, and farmers can make informed decisions about their livestock management, such as identifying low-performing cows and taking strategic steps to optimize their herd sizes (Borchers *et al.*, 2017) (Ushikubo *et al.*, 2017).

The problem of dairy production (income) estimation can be stated as a multivariate time series prediction task, based on multiple dairy features recorded periodically over the lactation months (on test days). The dairy features are categorized into health, management, milk quality, milk value, farm environmental and seasonal factors.

The general field of time series prediction has captured the interest of many researchers in recent years, and numerous deep learning-based models have been developed. In this paper, a novel model for dairy series prediction is described. The model is designed to predict the milk income in future lactation by taking into account the historical information, in continuation of previous related works (Naghashi *et al.*, 2023) (Frasco *et al.*, 2020) about dairy profitability prediction. The forecast generated by our model can offer valuable assistance for dairy farmers to make informed decisions about their livestock, such as whether to retain or cull cows from their herd. Our model is based on the Transformer model (Vaswani *et al.*, 2017), with architectural modifications to account for the temporal relations within the sequence of lactation months and the dependencies between the sequences of dairy variables recorded on specific days during the lactation months. Overall, the following contributions to the state-of-the-art (SOTA) have been made :

— The first Transformer-based model to predict dairy production ;
— A novel Transformer design that features :

— The efficient capture of intra-series (temporal) and inter-series (spatial) correlations, with the model's prediction based on the weighted sum of both results ;
— The use of backcast encoder connections to help each processing block focus on the essential portion of its input sequence.

Predicting the milk income during the upcoming lactation months involves tackling a multivariate time series prediction task, leveraging historical lactation data and pertinent dairy features. This type of forecasting requires the analysis of sequential data to discern trends and patterns, facilitating the learning process to predict future values of one or more variables of interest.

5.1.1 Related Work

In more recent years, the dairy industry has witnessed a proliferation of deep learning-based approaches. For instance, the Long-Short Term Memory (LSTM) neural network has been exploited for cow's milk profitability prediction (Frasco *et al.*, 2020). It relies on an integrated multiple source test-date information to predict the profits in future months. A deep learning framework is proposed in (Liseune *et al.*, 2021) to encode the dairy features, then predict the latent representation of milk yield sequence and generate the lactation curve by leveraging Convolutional Neural Networks (CNN) and encoder-decoder modules. The prediction of first test day milk yield is accomplished through the application of classical machine learning models, including random forest in (Salamone *et al.*, 2022). This is achieved by giving the models test day production, reproduction and herd features. Models for time series forecasting can be generally categorized into conventional statistical models and deep learning models. Statistical models, such as ARIMA (Autoregressive Integrated Moving Average) has shown a good performance (Contreras *et al.*, 2003). ARIMA corresponds to a class of linear models and its main shortcoming is that it is not applicable to non-linear systems and is a univariate model, not capable of utilizing other covariates in the prediction process. (Zhang *et al.*, 1998). One of the well-known deep learning models called N-BEATS was proposed for univariate time-series forecasting (Oreshkin *et al.*, 2019). It is based on backward and forward residual connections and a deep stack of fully-connected layers which is fast to train and is interpretable thanks to the trend and seasonality basis components. In order to capture short and long term temporal information in time series, LSTNet was proposed (Lai *et al.*, 2018), which exploits Convolutional Neural Network and Recurrent Neural Network to capture short term local dependencies between the variables and long term patterns of time series trend.

The Transformer architecture has shown promising performance in sequence processing and time series prediction, and many time series forecasting models have been proposed that utilize with some modifications. For instance, the work in (Wu *et al.*, 2020a) designed a deep Transformer model that includes both the encoder and decoder modules to forecast Influenza prevalence. The Informer, a variant of Transformer model adapted for time series forecasting introduced in (Zhou *et al.*, 2021), exploits a specific attention mechanism called "probSparse self-attention" along with a distillation mechanism. This hybrid approach emphasizes dominant attention scores and reduces computational overhead and memory consumption, making Informer well-suited for efficient processing of long time series data. The Autoformer (Wu *et al.*, 2021) and FEDFormer (Zhou *et al.*, 2022) utilize the Fourier transform to extract frequency information and combine it with time domain features inside the Transformer architecture. Recently patch-wise encoding has been most popular and proven to be useful in capturing temporal relations in long time series by using Transformer architecture. The Crossformer model (Zhang et Yan, 2022) captures cross-time and cross-dimension dependency by utilizing Dimension-Segment-Wise embedding and a two-stage attention mechanism to capture the correlations between the input series and the dependencies along the time

dimension. Another well-known model based on patch-wise attention is the PatchTST (Nie *et al.*, 2022) which uses channel independence and patch-level attention for data mining, however it doesn't capture the relations between different variables. More recently, (Liu *et al.*, 2023) proposed iTransformer which utilizes embedding along time dimension and then extracts the dependencies between variables through self-attention and feed-forward modules.

Nevertheless, both temporal and spatial (inter-series) dependencies can be extracted independently and then integrated into the final prediction. Moreover, connecting the layers in a residual fashion helps the next layer to focus on the more relevant signal portion. In this work, we have taken into account these aspects in order to enhance the performance of the Transformer model for prediction of dairy time series.

5.1.2 Transformer and multi-head self-attention module

The Transformer model has proven its efficiency in capturing long-term dependencies in a sequence (sentence or time series), thanks to its self-attention block which provides the correlations between different positions in a sequence. This allows to determine the importance of each input word or token to generate the output representation, and focus on the relevant elements. Moreover, a multi-head attention mechanism allows to process the input sequence from several viewpoints, enabling the model to capture different types of relationships (in different spaces) in the input sequence. To compute the multi-head self-attention, the input sequence stack is first converted into so called queries, keys, and value matrices through linear projections :

$$Q_i = X \cdot W_i^Q, \quad K_i = X \cdot W_i^K, \quad V_i = X \cdot W_i^V \tag{5.1}$$

In the above formulas, X represent the embedded input sequence, Q_i, K_i, and V_i are the projection results, i.e., the query, key, and value matrices, and W_i^Q, W_i^K, and W_i^V are the corresponding projection matrices for ith head. These matrices essentially map the input tokens into a common space where they can be compared. Each projection matrix is of size $d_{model} * d_k$, where d_{model} is the input dimension (embedding dimension) and d_k is the head dimension. The input multivariate time series is represented in a tensor including the samples with the variables (series values) over the input time steps. Thus, X is of dimension $(Batch, Length, d_{model})$. In this context, "batch" is referred as a collection of multiple samples, "Lengths" corresponds to the length of input time-series, and d_{model} signifies the dimension of the feature space in which the attributes are projected.

The head dimension is calculated as $d_k = d_{model}/h$, where h is the number of heads. The projected queries and keys are multiplied, and the result is scaled down by d_k before feeding it to a softmax function in order to obtain the attention scores :

$$Attention(Q, K, V) = softmax\left(\frac{Q \cdot K^T}{\sqrt{d_k}}\right) \cdot V \tag{5.2}$$

Dividing the dot products of queries and keys by the square root of the head dimension ensures stability during training and controlling the scale of the attention scores.

As mentioned, the input undergoes h projections into head space, each time with an independent self-attention computation :

$$head_i = Attention(XW_i^Q, XW_i^K, XW_i^V) \tag{5.3}$$

Then, the results are concatenated to represent the temporal dependencies as seen by the different attention heads, hence providing valuable information to the downstream blocks or tasks. Finally, the concatenated attention scores are linearly projected by a W^O matrix of size $d_k * d_{model}$ to convert them into an output that can be processed by additional transformer layers :

$$MultiHead(Q, K, V) = Concat\,(head_1, \ldots, head_h) \cdot W^O \tag{5.4}$$

5.2 Methods

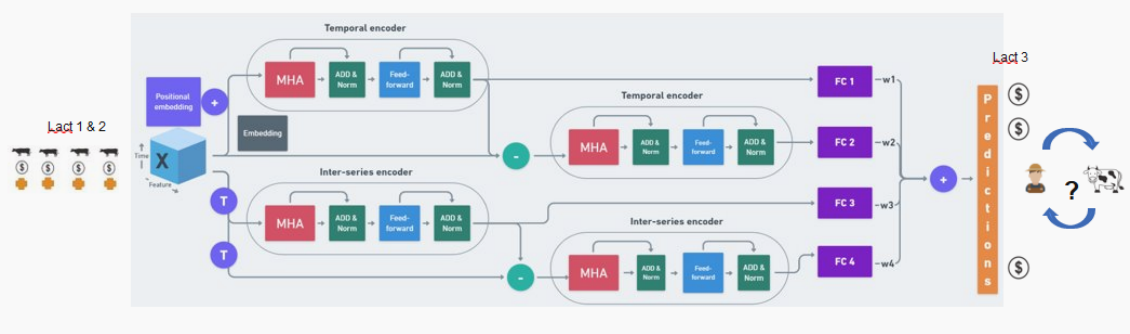


Figure 5.1 – The proposed Transformer based model with inter-series (spatial) self-attention modules. The input of the temporal encoder part is embedded and positional encoded in order to inject the sequence order information to the input series. The second temporal and spatial encoders take the residual inputs, which are obtained by subtracting the output of the previous block from its input. This helps the next encoder block to focus on the relevant part of the time series signal, which is achieved by subtracting the previously encoded information from the input signal, passing only the residual part to the next block for further utilization in the prediction process. The final prediction is the result of a weighted sum of the predictions obtained from each encoder after flattening and passing through the separate linear layers.

5.2.1 Problem definition

The problem of milk production prediction can be stated as follows : given a herd of cows of size N, each cow i represented by a set of records $\{\mathbf{x}_j^{(i)} = (x_{j,1}^{(i)}, ..., x_{j,T}^{(i)})\}_{j=1..v} \epsilon\, \Re^{v \times T}$, where $\mathbf{x}_j$ is a sequence of dairy variables, v represents the number of dairy variables considered and T is the record length (total length of the first and second lactation periods, fixed for all of the animals), predict M steps (months) of the upcoming

milk production in the third lactation : $\hat{r}^{(i)} \epsilon \Re^{M}$. Thus, the goal is to learn a non-linear function f which, given the input multivariate time series $X \epsilon \Re^{N \times T \times v}$, estimates milk income values in the future lactation months $\hat{r} \epsilon \Re^{N \times M}$, with the N cow samples presented to the model as the training set.

5.2.2 Inter-series correlations

The multiple series related to the different source variables impact each other and are correlated (Zhang et Yan, 2022). Therefore, capturing those dependencies is important for better forecasting. To achieve accurate results, it is essential to explicitly model the intricate relationships between the variables. The existing time series forecasting methods often fall short in capturing these complex inter-dependencies, leading to sub-optimal predictions. However, the self-attention mechanism can also capture inter-series dependencies, considering the temporal information (Liu *et al.*, 2021a). In this case, instead of calculating the attention scores at each time-step, the correlations between the series can be captured by calculating the attention importance scores for each pair of variables. For this purpose, the input multivariate time series is transposed before feeding it to the multi-head self attention module. The similar steps, including projection, attention score calculation is performed using the time dimension interchanged with the variable (series) dimension in order to obtain the importance scores for each pair of the variables. Therefore, the input to the inter-series (spatial) self-attention module is the transposed input sequence with the shape of $(Batch, Feature, Length)$ without adding any positional information.

It is important to mention that the inter-series self-attention module receives the transposed input multivariate series without embedding. This approach is chosen deliberately to maintain low time complexity and preserve the original number of input time-steps, because the embedding changes the number of time-steps when applied on the time dimension. In the projection formulas below, X is the transposed input multivariate series without embedding. W^{Q}_{int}, W^{K}_{int} and W^{V}_{int} are of dimensions $d_T * d_T$ with d_T indicating the time dimension size (number of the input time steps).

$$Q_{int} = X \cdot W^{Q}_{int}, \quad K_{int} = X \cdot W^{K}_{int}, \quad V_{int} = X \cdot W^{V}_{int} \tag{5.5}$$

The inter-series attention scores are calculated in the same way as in the standard self-attention :

$$Score_{int} = softmax\left(\frac{Q_{int} \cdot K^{T}_{int}}{\sqrt{d_T}}\right) \tag{5.6}$$

$$Attention(Q_{int}, K_{int}, V_{int}) = Score_{int} \cdot V_{int} \tag{5.7}$$

In order to keep the computational complexity low, multi-head attention is reduced to single head self-attention in the inter-series attention module.

5.2.3 Residual Blocks

The residual blocks within the N-BEATS (Oreshkin *et al.*, 2019) model plays a crucial role in enhancing prediction accuracy across successive blocks. These blocks facilitate the selective transfer of relevant information, allowing only essential signals to be forwarded to the subsequent blocks. As a result, each subsequent block can focus and build upon the remaining pertinent details passed down from the previous block, contributing to an overall improvement in the model's forecasting capabilities. Inspired by the forecast and back-cast outputs design, the successive encoder blocks in our model are connected using the back-cast inter-connection, which is the residual part of the latest encoder output obtained by subtracting the output of the previous encoder module from its input.

5.2.4 Proposed model

The architecture of the proposed Transformer based model is illustrated in Figure 5.1. This model exploits the power of the self-attention mechanism to extract meaningful temporal and spatial (inter-series) correlations. Two separate Transformer encoder blocks are the main building components. The first encoder block consists of the multi-head self-attention and feed forward components connected to each other along with residual connections. Each component is followed by a normalization layer. This prevents covariance shift across layers during training, which may cause gradient problems and their negative effect on convergence. The second parallel encoder block has the same structure, except the input sequence, where the original input time series is transposed to swap the time and feature dimensions. Then, the ensuing attention scores indicate the pairwise inter-variable dependencies at each time step. In this paper, the network is of depth 2, since two successive parallel temporal and series-wise encoder blocks constitute the model. More layers can be added depending on data availability and dataset attributes.

In the proposed model, all the encoder outputs contribute to the final prediction. The encoders in the first layer extract shallow features (both temporal and inter-series ones), while the encoders in the second layer capture more complex patterns. Therefore, combining all those outputs contribute to obtain better results. The encoder outputs are flattened and fed into separate linear layers (fully-connected with no activation function) for producing separate predictions, with different outputs contributing to a weighted sum that gives the final prediction :

$$\text{Prediction} = \sum_{i=1}^{4} w_i \cdot o_i \tag{5.8}$$

In which o_i is the prediction from the ith encoder and w_i is the associated weight. The use of a weighted sum for the final prediction allows to account for the relative importance of the different outputs that contribute to it, which should lead to better forecasts and facilitates the interpretation of the final prediction. The weights are learned like any other network parameters in an end-to-end manner. They are randomly initialized and then normalized using the softmax function. We called our model Transformer-BEATS as it exploits the power of the Transformer (Vaswani *et al.*, 2017) blocks and the back-cast or residual connections introduced in the N-BEATS model (Oreshkin *et al.*, 2019) in an adaptive architecture.

5.3 Results

The selected dairy dataset for training consists of the records of 147,749 dairy cows from 5,844 Canadian dairy farms, recorded on test days during the first and second lactation periods from 2006 to 2017 (Naghashi *et al.*, 2023). Each record includes metrics of milk quality, seasonality, year, health, and management factors. The dataset was curated from 1.48 million animals after animal selection, outlier removal and missing data imputation. More information on the dataset can be found in the appendices section.

5.3.1 Experimental settings

The dataset is randomly split into 100,000 training and 47,749 test samples. The length of the input sequence is set to 22 since the model uses the multivariate series representing the first and second lactation periods to predict the milk values (income) during the third lactation months (11 steps ahead). The Mean Absolute Error (MAE), Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) are selected to assess the sequence prediction performance of our model and the baselines. These error measures are calculated based on the difference between the predicted milk income (in Canadian dollars) and the ground truth values over the upcoming 11 months.

We compared our Transformer-BEATS model to eight other time series prediction and sequence modeling approaches, including Auto-ARIMA (Contreras *et al.*, 2003), MuMu (Frasco *et al.*, 2020), Vanilla Transformer (Vaswani *et al.*, 2017), N-BEATS (Oreshkin *et al.*, 2019), Crossformer (Zhang et Yan, 2022), DLinear (Zeng *et al.*, 2023), PatchTST (Nie *et al.*, 2022) and iTransformer (Liu *et al.*, 2023). More details about these baseline models can be found in Related Work section.

5.3.2 Main results

Table 5.1 represents the experimental results related to the dairy prediction, showcasing various error metrics computed across the target 11-month lactation period. The results highlight the superior performance of our Transformer-based model, surpassing other models, particularly outperforming all variations of Transformer models and the recently proposed linear model (DLinear). Here also, the proposed Transformer model obtained more similar predictions to the ground truth, compared to the other methods, specifically compared with the PatchTST model. The reason is that PatchTST ignores the intricate relations between the variables and the interactions among them. While our Transformer-BEATS model, similar to the iTransformer, takes those relations into consideration. Furthermore, the iTransformer model, which has been recently proposed, neglects the temporal dimension of time series data, falling short in capturing long-term temporal correlations. The prediction results and visualization of the learned attention maps (both temporal and spatial attentions) show the advantage of combining the long-term temporal correlations with the inter-series patterns extracted from different encoder layers. By focusing on relevant time steps, the model can better understand the temporal dependencies and patterns present in the data, and the multiple layers of the inter-series encoders obtain spatial relations between the variables at different levels. In order to further analyze the effectiveness of our Transformer-BEATS, we divided the test sample series into tightly correlated and low correlated samples using the average Dynamic Time Warping (DTW) distance between their associated variables. Our model gains RMSE error improvements of 5% over the PatchTST model which doesn't deploy a mechanism to capture the interactions between the dairy variables. The prediction result of our model is also competitive and even better that those of the Crossformer and iTransformer models

Table 5.1 – Comparison of baseline methods with the proposed approach on the dairy production dataset in terms of cumulative Absolute Error, cumulative Root Mean Square Error, Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE).

Category	**Model**	**RMSE of Cumulative Milk**	**MAE of Cumulative Milk**	**RMSE**	**MAE**	**MAPE**
Univariate	ARIMA	52.675	42.342	7.313	5.769	27.197
	N-BEATS	32.212	24.974	4.198	3.257	17.479
Multivariate	Transformer	30.183	23.417	4.033	3.126	16.553
	MuMu	30.014	23.276	4.016	3.110	16.502
	DLinear	31.451	24.489	4.145	3.221	17.928
	Crossformer	30.039	23.288	4.017	3.111	16.454
	PatchTST	32.452	25.333	4.213	3.283	17.411
	iTransformer	30.564	23.718	4.069	3.156	16.772
	Transformer-BEATS	**29.947**	**23.208**	**4.004**	**3.100**	**16.448**

which exploit attention over the channel dimension to represent inter-series correlations. In addition, the results in terms of the error metrics in Table 5.1 pinpoint the promising performance of our model in dairy prediction of the sample dairy series with inter-variable correlations. Certain milk quality factors exhibit higher correlations, notably milk, fat, and protein yield (Schaeffer et Jamrozik, 1996). Through our model's analysis, these dependencies are effectively highlighted as evidenced by the higher attention scores assigned to the corresponding feature pairs (milk, fat, and protein yields) in the inter-series attention map learned by our model. High attention scores are also assigned to DIM (Days in milk)-Lactose and DIM-FTP (Fat to Protein ratio) feature pairs, because the FTP and lactose amount are usually high during the early lactation months and as the lactation period progresses and the cow moves further into mid and late lactation, the lactose and fat to protein ratio tends to decrease (Lim *et al.*, 2020).

5.4 Discussion

We further evaluate the spatial (inter-series) relations by inspecting the attention scores through visualization. Figure 5.3 shows the mean spatial attention map for the test samples (the learned attention map after training the model) together with the mean dynamic time warping (DTW) distance across the time steps

Table 5.2 – Results of the ablation study (We run all the experiments three times with different seeds to reduce the randomness effect). The best results are indicated in bold text and the second best ones in italics.

Transformer components	Masked Attention	RMSE	MAE	Cumulative RMSE	Cumulative MAE
Backcast + Learned Weights	Yes	*4.004*	**3.099**	*29.952*	*23.205*
Backcast + Learned Weights	-	4.005	3.102	29.984	23.244
Inter-Series + Backcast + Learned Weights	Yes	**4.003**	*3.100*	**29.947**	23.208
Inter-Series + Backcast + Learned Weights	-	4.009	3.105	30.022	23.273
Inter-Series + Learned Weights	Yes	4.007	3.104	29.976	23.232
Inter-Series + Learned Weights	-	4.005	3.100	**29.947**	**23.189**
Inter-Series + Backcast + Fixed Weights	Yes	4.014	3.109	30.038	23.289
Inter-Series + Backcast + Fixed Weights	-	4.012	3.108	30.045	23.282

of different channels. Three different blocks are selected in both attention map and the DTW distance matrix. The mean raw time series of the corresponding channels are plotted in Figure 5.2 to show the relation between the spatial attention scores and the similarity between the corresponding raw series (trend and shape).

According to Figure 5.3 (left figure) a1 block has a relatively high spatial attention score, which means PT-Milk Yield have similar trend and shape. Looking at Figure 5.2 top, we can observe that PT (Protein yield) and Milk Yield series are very similar in terms of the trend and shapelet. Meanwhile, blocks a2 and a3 indicate very small attention scores, and the middle and bottom plots in Figure 5.2 are the most clear evidence, by showing significantly different trends associated with the DIM and Milk Yield series (bottom sub-figure) and very different shapes between DIM and SCC (Somatic Cell Count) series (middle subfigure). Note that DTW distance matrix (Figure 5.3 right) represents very small distance between PT-Milk Yield (block b1), proving that the learned spatial attention scores could capture those types of similarities among the dairy series. Based on the DTW matrix (Figure 5.3 right), it's clear that the distance between DIM-SCC series (block b2) and DIM-Milk Yield (block b3) exhibit relatively higher values. Upon inspecting the associated attention map (blocks a2 and a3), it becomes apparent that their associated learned attention scores are notably low, as indicated by comparing their related series plots. This implies that our model has effectively discerned the disparity in trends between the DIM and Milk Yield series and the difference in shapes between DIM and SCC series according to the insights provided by the DTW measures. We derived the weight vector [0.0345, 0.1198, 0.6469, 0.1988] learned through the training of our model. These weights are assigned to the predictions of the first and second temporal encoders, as well as the first and second spatial encoders within the model, respectively. This allocation highlights the significant contribution of the first spatial encoder block (weight = 0.6469), emphasizing its crucial role in cross-channel representation.

5.4.1 Running time and performance analysis

In this subsection, the training time of each model in one epoch is calculated and plotted against its corresponding error metric (Figure 5.4). Based on this figure, our Transformer-BEATS model has the lowest error but its running time is higher than recurrent neural network based model (MuMu) and iTransformer. Our model is much faster than other Transformer based models recently proposed for time series forcasting (PatchTST and Crossformer) while providing more accurate results in the task of dairy series forecasting.

5.4.2 Ablation study

We conducted an ablation study on the designated dairy dataset to evaluate the efficacy of the different components incorporated in our proposed model. Each configuration was executed three times, and the averaged results are presented in Table 8.7. The ensuing observations provide insights into :

- Employing masking within the self-attention module, a technique applied in the basic Transformer model to ensure that predictions for individual positions solely rely on the preceding positions (causality), leads to slightly better prediction performance in most cases.
- Removing the inter-series self-attention encoders increases the cumulative RMSE error of the target lactation period, showcasing the importance of the inter-series encoders.
- Utilizing back-cast connections combined with the causal masking yields better outcomes compared to their absence.
- The learned weights enable the model to generate predictions by assigning significance to various outcomes in a data-driven manner and the learned weights are interpretable by showing the contribution of each component (encoder block). After establishing these weights (to fixed weights of 0.25), we conducted a replicated experiment to observe the influence of these adaptive weight assignments. The result indicates that our model with learnable weights, outperforms the same architecture set by the fixed weights.

5.4.3 Ablation on longer horizon prediction and other domains

We selected some public datasets from various domains, including traffic, weather, solar-energy and electricity for long term forecasting to conduct experiments using our Transformer-BEATS and the other baseline models. The results pinpoint the efficiency of our model when applied to other datasets and use cases according to Table 5.3. We should mention that our model is based on channel-mixing, which embeds all of the channels to the model dimension. In this experiments, the input series is divided into patches of equal length with a specific patch size and stride (overlapping region between two consecutive patches). Then, the patches are mapped into the model dimension and after adding the positional encoding are passed to the temporal encoder of the Transformer-BEATS model and the spatial encoder receives the transposed version of the original series without patching applied on it.

Additional experiments were conducted to indicate the performance of our model on different forecast horizons. The models were trained using a shorter input window and tested on longer prediction horizon (two lactation periods instead of one). Transformer-BEATS model achieved the best results in terms of the error metrics in comparison to the other baseline models according to Table 5.4. Our model predicts the milk income with 4.7% and 5.5% of Cumulative RMSE reduction compared to the iTransformer and PatchTST

Table 5.3 – Comparison of models on public datasets in terms of Mean Squared Error (MSE) and Mean Absolute Error (MAE) across different forecasting horizons.

Models		Transformer-BEATS		iTransformer		PatchTST		Crossformer		FEDformer		Autoformer	
Dataset	Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ECL	96	*0.153*	*0.256*	**0.148**	**0.240**	0.195	0.285	0.219	0.314	0.193	0.308	0.201	0.317
	192	*0.172*	*0.270*	**0.162**	**0.253**	0.199	0.289	0.231	0.322	0.201	0.315	0.222	0.334
	336	*0.193*	0.289	**0.178**	**0.269**	0.215	0.305	0.246	0.337	0.214	0.329	0.231	0.338
	720	*0.227*	**0.314**	**0.225**	*0.317*	0.256	0.337	0.280	0.263	0.246	0.355	0.254	0.361
PEMS07	12	**0.059**	**0.155**	*0.067*	*0.165*	0.095	0.207	0.094	0.200	0.109	0.225	0.199	0.336
	24	**0.083**	**0.183**	0.088	0.190	0.150	0.262	0.139	0.247	0.125	0.244	0.323	0.420
	48	*0.120*	*0.226*	**0.110**	**0.215**	0.253	0.340	0.311	0.369	0.165	0.288	0.390	0.470
	96	*0.181*	*0.283*	**0.139**	**0.245**	0.346	0.404	0.396	0.442	0.262	0.376	0.554	0.578
Weather	96	*0.164*	**0.214**	0.174	0.214	0.177	0.218	**0.158**	0.230	0.217	0.296	0.266	0.336
	192	*0.215*	0.260	0.221	**0.254**	0.225	*0.259*	**0.206**	0.277	0.276	0.336	0.307	0.367
	336	**0.271**	*0.297*	0.278	**0.296**	0.278	0.297	*0.272*	0.335	0.339	0.380	0.359	0.395
	720	0.382	0.366	0.358	*0.349*	*0.354*	**0.348**	**0.351**	0.386	0.403	0.428	0.419	0.428
Solar-Energy	96	**0.201**	*0.249*	*0.203*	**0.237**	0.234	0.286	0.310	0.331	0.242	0.342	0.884	0.711
	192	**0.233**	*0.271*	*0.233*	**0.261**	0.267	0.310	0.734	0.725	0.285	0.380	0.834	0.692
	336	*0.250*	*0.284*	**0.248**	**0.273**	0.290	0.315	0.750	0.735	0.282	0.376	0.941	0.723
	720	*0.273*	*0.309*	**0.249**	**0.275**	0.289	0.317	0.769	0.765	0.357	0.427	0.882	0.717

models, respectively, indicating its capability in handling longer horizon prediction of dairy production.

5.5 Conclusion

We propose Transformer-BEATS, a Transformer based framework for prediction of the multivariate time series of the dairy milk income in upcoming lactation months. Our model captures both temporal and cross-channel dependencies in the dairy time series data by combining different Transformer encoders, each focusing on a specific task of capturing temporal or inter-series relations. Further, those module are inter-connected in a residual (back-cast) manner to help useful signal propagation to the next block. Our experiments on a large benchmark dairy dataset have shown that our proposed model outperforms several classical and state-of-the-art models.

5.6 Appendix : Dairy dataset and pre-processing

The benchmark dairy dataset used to evaluate our multivariate time series model and the selected baseline methods consists of a set of dairy attributes that include metrics of milk quality, seasonality, year, health, and management factors, recorded during the first, second and third lactation cycles for 147,749 dairy cows from 5,844 Canadian dairy farms during the years of 2006 to 2017. More precisely, the dataset includes the following features : 1- HR-24-Milk (Milk yield in 24 hours), 2- Milk Value or Income (in CAD), 3- Somatic Cell Count (SCC as a health indicator), 4- Milk Urea Nitrogen (MUN is a health factor), 5- LACTOSE (Lactose yield), 6- FAT (Fat yield in 24 hours), 7- PT (Protein yield in 24 hours), 8- DIM (Days in Milk), 9- FTP (Fat to Protein ratio), 10- Lactation number, 11- MILK-FQCY (Number of milkings per day), 12- MILK-PTRN (Milking time in A.M, P.M, both), 13- Test season, 14- Birth season, 15- Animal condition, 16- Test year, 17- Test month. The prediction target series (variable) is the monthly income from milk sales (Milk Value in Canadian dollar

Table 5.4 – Forecasting results related to longer horizon prediction in the dairy production dataset (Input sequence length = 11 months and target sequence length = 22 months).

Method	Cum-RMSE	Cum-MAE
Transformer-BEATS	**54.417**	**42.716**
DLinear	56.978	44.783
MuMu	55.024	43.198
iTransformer	57.118	44.814
Crossformer	54.623	42.780
PatchTST	57.613	45.666

(CAD)) measured at each cow during their third lactation. More information regarding the dairy factors (variables) can be found in Table 5.5.

5.7 Appendix : Public datasets

Experiments are conducted on 4 complex real-world datasets to evaluate the performance of the proposed Transformer-BEATS model including (1) Weather (Wu *et al.*, 2022) composed of 21 meteorological factors collected every 10 minutes from the Weather Station of the Max Planck Biogeochemistry Institute in 2020. (2) ECL (Wu *et al.*, 2022) consists of the hourly electricity consumption data of 321 clients. (3) Solar-Energy (Lai *et al.*, 2018) records the solar power production of 137 PV plants in 2006, sampled every 10 minutes. (4) PEMS07 is recorded based on the public traffic network data in California collected by 5-minute windows. The details are shown in Table 6.1.

Table 5.5 – Summary of numerical dairy variables used in the framework.

Variable	**Missing %**	**Min**	**Max**	**Mean**	**Std**
Milk Yield (Kg/24h)	1.4	0.20	61.40	29.60	9.25
Milk Value or Income (in Canadian Dollars (CAD))	0.002	0.09	42.95	21.21	5.94
Somatic Cell Count (1000/mL)	17.62	1	1754	156.59	254.87
Milk Urea Nitrogen (MUN)	56.04	2.10	19.80	10.91	3.27
Lactose Yield (Kg)	37.63	0	6.30	4.55	0.22
Fat Yield (Kg/24h)	1.64	0	2.40	1.19	0.38
Protein Yield (Kg/24h)	1.64	0	1.94	1.00	0.30
Days in Milking (DIM)	0	0	517	169.56	99.39
Fat to Protein Ratio (FTP)	1.64	0.73	1.67	1.20	0.16

Table 5.6 – Detailed dataset descriptions. Dim denotes the variate number of each dataset. Dataset Size denotes the total number of time points in (Train, Validation, Test) split, respectively. Frequency indicates the sampling interval of time points.

Dataset	Dim	Prediction Length	Dataset Size (Train, Validation, Test)	Frequency	Field
ECL	321	96, 192, 336, 512	(18317, 2633, 5261)	Hourly	Electricity
PEMS07	883	12, 24, 48, 96	(16911,5622,468)	5 min	Traffic
Weather	21	96, 192, 336, 512	(36792, 5271, 10540)	10 min	Weather
Solar Energy	137	96, 192, 336, 512	(36601, 5161, 10417)	10 min	Energy

5.7.1 Dairy dataset pre-processing

Similar to (Naghashi *et al.*, 2023), the following steps were taken to pre-process the benchmark dataset :

- Retaining only animals with test records for the first, second, and third lactations.
- Omitting records from the dry period, i.e. the months during which a cow does not produce milk, which typically occurs between lactation cycles, with a near-zero milk value (income) as a result.
- Excluding animals that left the herd before or during the third lactation.
- Removing duplicate records.
- Eliminating records with negative milk value and cumulative milk value : Although these cases had constituted a minor fraction of the dataset, they were removed to avoid inconsistencies. For instance, negative values of dairy income (expressed in CAD), may be the result of data acquisition errors or exogenous expenses such as the acquisition of new equipment or repairs.
- Deleting records with contradictory data, such as rows implying an active milking phase but featuring zero milk yield.
- Removal of outliers (their respective rows) : Outlier records are those with more than ± 2.5 × Standard Deviation range for the mean of one (or more) dairy attribute.
- Missing data imputation : Using the herd ID, season and year (grouping the animals and using mean of each group to replace the missing values that lie in the same group).

From the 147,749 dairy cows, we randomly selected 100,000 to train the models and the remaining 47,749 for evaluation. The missing data was imputed after train-test split to avoid information leakage (Naghashi *et al.*, 2023). The categorical variables are transformed using one-hot encoding, while the numerical variables (series) are standardized by taking into account their mean and standard deviation across both batch and time dimensions. Then, they are merged as the final multivariate series (52 series in total).

5.8 Appendix : Experimental setting

We used the Grid Search method for hyper-parameter optimization (Zhou *et al.*, 2021). We investigated a set of possible values for each hyper-parameter and trained the model using it. For example, different model dimensions were employed, and the one leading to the lowest prediction error during the testing time was selected in the final model configuration. The same process is performed for other hyper-parameters, including learning rate, batch size, optimizer type and number of train epochs.

In experiments regarding public datasets, we trained the models using MSE (Mean Square Error loss) for

10 epochs using Adam optimizer and the early stopping policy (patience number set to 3). We divided the input series into patches of the same size (length of 8) according to (Nie *et al.*, 2022) which reduces the computational cost and helps the model to capture the long-term temporal correlations. We should mention that Transformer-BEATS (ours) takes channel-mixing in treating time series representation (it maps the whole channels to the model dimension).

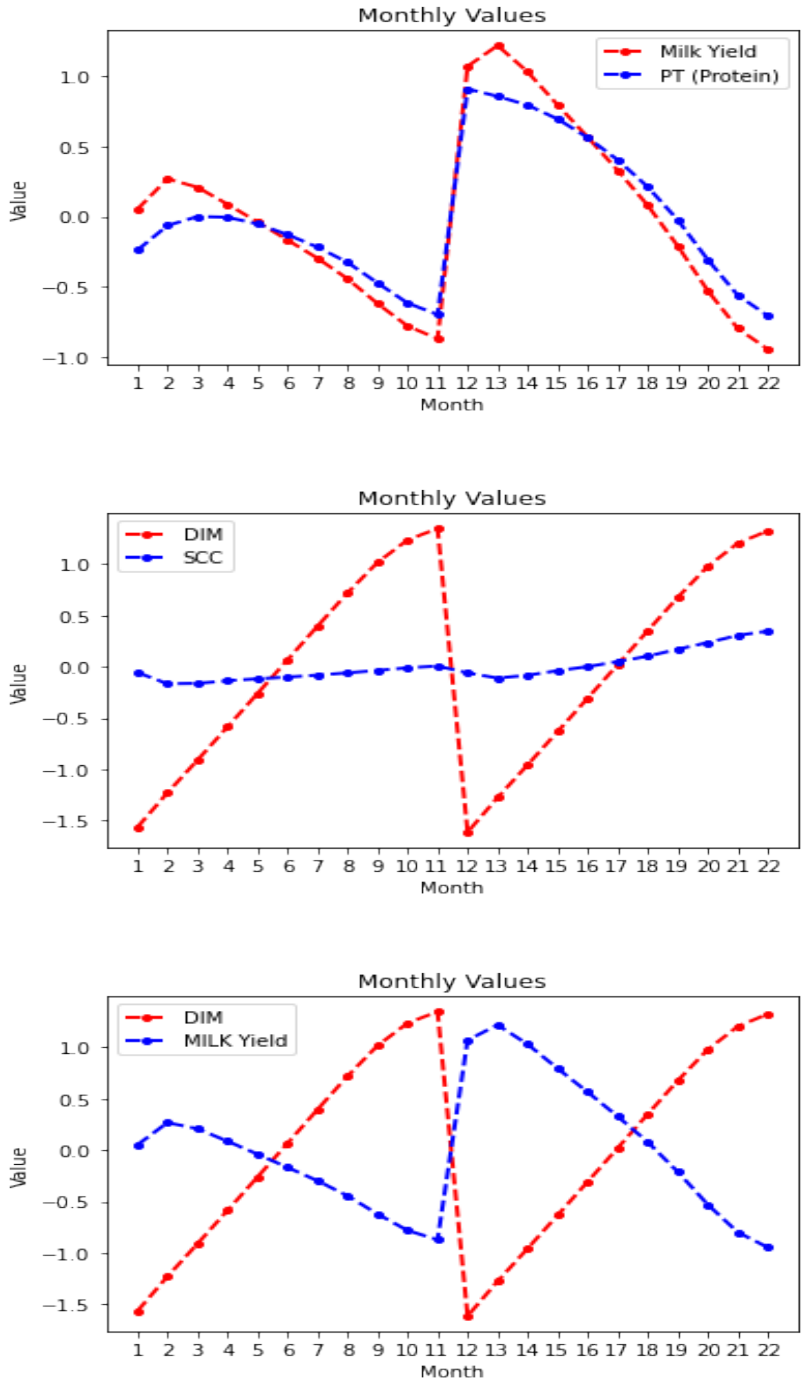


Figure 5.2 – Average normalized time series (Top : Mean Protein and milk yield, Middle : Mean DIM (Days In Milk) and SCC (Somatic Cell Count), Bottom : Mean DIM and milk yield).

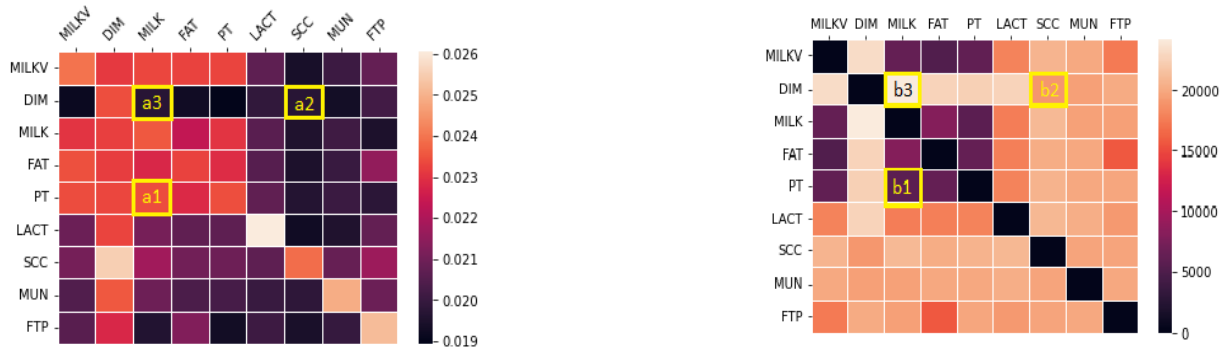


Figure 5.3 – Left : Mean inter-series (spatial) attention map learned from the data samples (numerical features). Right : Mean channel-wise dynamic time warping distance matrix of the test samples. Blocks *b1*, *b2*, and *b3* indicate low, high, and relatively high DTW distances between dairy feature pairs, respectively. Similarly, blocks *a1*, *a2*, and *a3* represent high, low, and low attention scores, respectively. A low DTW distance corresponds to a high attention score and vice versa.

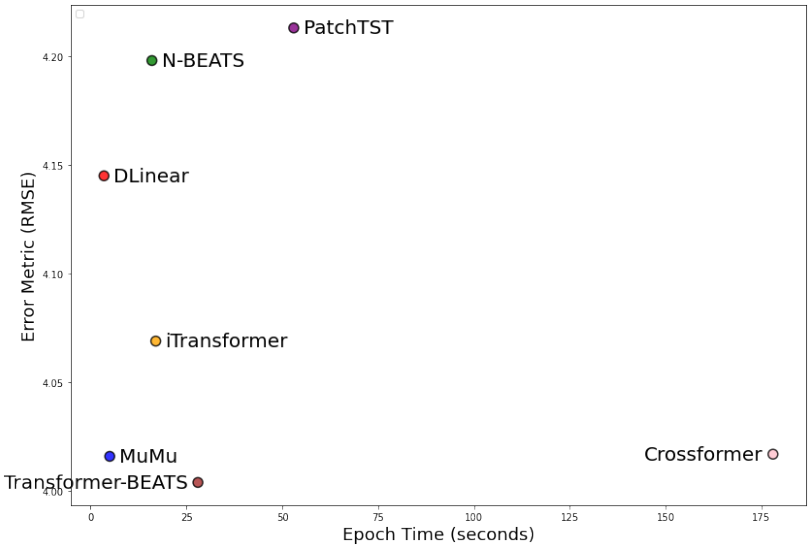


Figure 5.4 – Relationship between training epoch time (in seconds) and error metric (RMSE) for each model.

CHAPITRE 6
A MULTISCALE MODEL FOR MULTIVARITE TIME SERIES FORECASTING

This chapter is based on the work presented in my paper titled "MultiPatchFormer : A Multi-Scale Transformer for Multivariate Time Series Forecasting". In this work, I address key limitations of existing Transformer-based forecasting models, particularly their tendency to operate at a single temporal scale and overlook cross-series dependencies. The proposed model, MultiPatchFormer, introduces a novel architecture that combines multi-scale patch-wise encoding with a channel-wise attention mechanism to better capture both short- and long-term dependencies across multiple time resolutions. A multi-step linear decoder is further employed to enhance generalization and mitigate noise effect. In addition, the cross-channel encoder, addresses the limitations of previous Transformer based models in representation of cross-variable correlations. The model is extensively evaluated on seven real-world datasets using different metrics and achieved state-of-the-art performance in terms of accuracy and robustness.

The work presented in this chapter was published in Scientific Reports under the title “A multiscale model for multivariate time series forecasting.” This study directly addresses and validates Hypotheses 3, 4, and 5 of this thesis. Article writing, conceptualization, model design, implementation, and experiments were performed by Vahid Naghashi, under the supervision of professors Abdoulaye Baniré Diallo and Mounir Boukadoum. A printed version of this paper is illustrated in Appendix C.

RÉSUMÉ

Transformer based models for time series forecasting have shown promising performance and during the past few years different Transformer variants have been proposed in time series forecasting domain. However, most of the existing methods, mainly represent the time series from a single scale, making it challenging to capture various time granularities or ignore inter-series correlations between the series which might lead to inaccurate forecasts. In this paper, we address the above mentioned shortcomings and propose a Transformer based model which integrates multi-scale patch-wise temporal modeling and channel-wise representation. In the multi-scale temporal part, the input time series is divided into patches of different resolutions to capture temporal correlations associated with various scales. The channel-wise encoder which comes after the temporal encoder, models the relations among the input series to capture the intricate interactions between them. In our framework, we further design a multi-step linear decoder to generate the final predictions for the purpose of reducing over-fitting and noise effects. Extensive experiments on seven real world datasets indicate that our model (MultiPatchFormer) achieves state-of-the-art results by surpassing other current baseline models in terms of error metrics and shows stronger generalizability.

6.1 Introduction

Time series forecasting plays an essential role in many fields, including finance, agriculture, meteorology and energy consumption domains. Motivated by its widespread application in different fields of Natural Language Processing and Computer Vision (Brown *et al.*, 2020) (Dosovitskiy, 2020), Transformer (Vaswani *et al.*, 2017) is exploited in various time series applications (forecasting, imputation and classification) with some modifications in its architecture (Zhou *et al.*, 2022) (Zhou *et al.*, 2021). While recent studies have raised doubts about the performance of Transformers, specifically when employing linear models with superior performance (Zeng *et al.*, 2023), the inherent capabilities of Transformers remain promising (Nie *et al.*, 2022) (Chen *et al.*, 2024), particularly in representing complex and non-linear datasets. This underscores the need for further modifications to fully harness their capability in the domain of time series forecasting.

Real-world multivariate time series exhibit high correlations between different variates and fluctuations at various temporal scales. For example, electricity consumption shows specific temporal variations spanning seasonal, daily and hourly granularities. Figure 6.1, illustrates a time series of a stock over one year, in which relations between patches of different scales are critical to capture more information regarding the local and global temporal dependencies from various perspectives. This calls for multi-scale modeling of time series (Ferreira *et al.*, 2006) and representation of inter series correlations (Liu *et al.*, 2023).

Most of the existing time series forecasting models lack an efficient mechanism for multi-scale representation and they totally rely on a single scale or time resolution. Although some of the baseline models consider multi-scale representation in their modeling process (Shabani *et al.*, 2022), (Chen *et al.*, 2024), however, these models employ separate sets of parameters for capturing temporal dependencies at each scale, which significantly increases time complexity and the risk of overfitting. Furthermore, most of the aforementioned methods, ignore the cross-channel relationships between time series channels which has been proved to be critical in time series analysis task (Liu *et al.*, 2023). Another limitation of the current research is the single step decoding of the encoded representation, particularly in dealing with long horizon prediction, which raises the risk of overfitting and makes the model more susceptible to be affected by noise.

To leverage the capabilities of Transformers for addressing the above mentioned issues, we aim to enhance the capture of both multi-scale and cross-channel dependencies, thereby aggregating this information for time series representation. We further divide input series into local patches to process the input time series and since the variable independence has been proved to be more efficient (Nie *et al.*, 2022), we feed the multivariate series to the Transformer blocks by keeping the variate (channel) dimension intact. We further divide the series into patches of different size and map patch length to the model dimension by using 1-dimensional convolution in order to leverage the local information inside a patch (intra-patch relations). Instead of using several Transformer blocks, we first embed the time series using different scales and aggregate them into the same model dimension (feature space). Then, we process the mapped series using a Transformer block, followed by an channel-wise Transformer component to capture cross-series dependencies. To further improve the Transformer for time series forecasting task, we introduce a channel dimensionality reduction method inside the multi-head self attention module when applied on the channel dimension, which helps to reduce over-fitting noise, specifically when dealing with the time series composed of many variates (channels). We additionally, devise a multiple step and pseudo auto-regressive decoding of prediction window by generating the predictions segment by segment using the encoded series and previously generated segment, in order to reduce the noise impact, particularly in long-term forecasting scenarios. The effectiveness of the proposed model is verified through different real-world benchmarks.

Here is the key summarization of our contributions :

1. We propose a time series forecasting model which effectively utilizes the multi-scale information and captures the inter-series dependencies among different channels.
2. Our model demonstrates superior performance on various time series datasets and shows robust performance in time series forecasting task with long input length.
3. We design a multi-scale time series embedding method which captures the local temporal information from various scales and divides time series to patches of different lengths, preparing it to represent long term temporal correlations from multiple temporal aspects.
4. We propose a pseudo auto-regressive (multi-step) decoding scheme to forecast the time series in multiple sequential steps rather than decoding the whole prediction length using a single linear layer in order to reduce the noise effect.

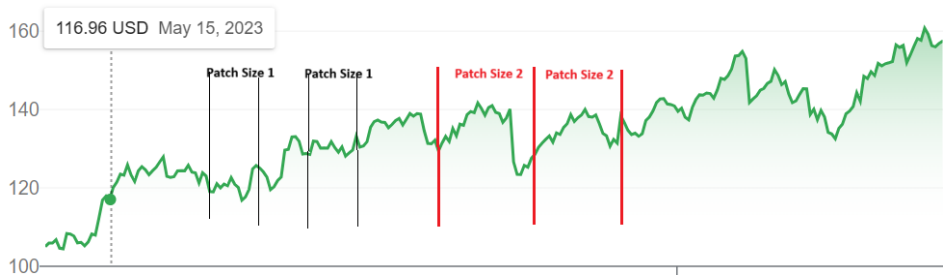


Figure 6.1 – Multi-scale dependencies in a real-world time series instance. The temporal patterns within patches of lengths Patch Size 1 and Patch Size 2 show similar trends and seasonality. Capturing these relations across different scales is crucial for analyzing time series data effectively.

6.2 Related Work

Time series forecasting involves predicting future values of one or multiple series based on the historical information. Time series forecasting methods are mainly categorized into classical and deep learning models. Among the statistical models, ARIMA (Elsaraiti *et al.*, 2021) and methods based on exponential smoothing (Hyndman et Khandakar, 2008) are well-known baselines for time series forecasting. The deep learning methods are based on Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN) and Transformer models. Among the RNN based model, DeepAR exploits RNN and auto-regressive decoding to predict the time series (Salinas *et al.*, 2020). CNN models use convolutions to extract temporal and sub-series dependencies (Wu *et al.*, 2022) (Liu *et al.*, 2022). For example, SCINet (Liu *et al.*, 2022) utilizes multiple convolutions to extract temporal information from different down sampled versions of the series. TimesNet (Wu *et al.*, 2022) on the other hand, converts the original one-dimensional time series into two-dimensional series and uses convolutions to capture inter-period and intra-period information. Some other linear models based on Multi-Layer Perceptron (MLP) have been proposed in (Das *et al.*, 2023) and (Zeng *et al.*, 2023) which exhibit an effective performance in time series domain. Several works have leveraged Graph Neural Networks (GNN) for time series forecasting, specifically for traffic forecasting. For example, MTGNN (Wu *et al.*, 2020b) utilizes temporal and graph convolutional layers to capture temporal and spatial (cross-channel) dependencies. An attention based spatial-temporal graph neural network is proposed in (Guo *et al.*, 2021) which captures the temporal dynamics of traffic series by using the local context. In (Wang *et al.*, 2022), a feature correlation-aware spatio-temporal graph convolutional network is designated for traffic prediction, which captures multi-scale spatial and temporal relations effectively, considering cross-scales dependencies. Dynamic graph structure learning for multivariate time series forecasting (Li *et al.*, 2023b) exploits graph learning networks to capture hidden dependencies between variables, enhancing the accuracy of forecasting by effectively capturing complex interrelationships within the data. Another work (Zhao *et al.*, 2023) addresses the problem of capturing dynamic correlations by learning historical relation graphs and predicting future relation graphs. They also design a causal GNN for feature extraction and reasoning network to capture the relations between historical time steps and forecasting horizon. Recent studies in time series analysis have also focused on reducing the computational and memory requirements related to processing large datasets. For instance, (Miao *et al.*, 2024) introduced TimeDC, a time series dataset condensation framework to preserve complex temporal relations while significantly reducing dataset size.

With the outstanding breakthrough in Natural Language Processing (NLP) and Computer Vision (CV) fields, Transformer models have recently shown superior performance in time series forecasting task and they

have been continuously evolving. Among them, Informer (Zhou *et al.*, 2021) develops a Transformer model based on prob-sparse self-attention to select important keys and reduce time complexity of self-attention. In Autoformer (Wu *et al.*, 2021), the self-attention is replaced with auto-correlation to capture temporal dynamics. FEDformer (Zhou *et al.*, 2022) utilizes Fourier transformer to deal with time series data given the fact that time series tend to have a sparse representation in Fourier basis. Recently, some linear models have been developed which outperform Transformer models in time series domain (Zeng *et al.*, 2023) and raised the concern about the efficiency of Transformer for time series forecasting. However, the PatchTST model (Nie *et al.*, 2022) proved the efficiency of Transformer models in time series analysis, which exploits patching and channel independence to model time series data, pinpointing that Transformer architecture is still a powerful model with some adaptation and architectural adjustments. Following PatchTST, other Transformer models have been developed for time series and proved high capability in dealing with high-dimensional time series (Liu *et al.*, 2023). Multi-scale representation proved to be necessary for time series analysis (Cirstea *et al.*, 2022) (Shabani *et al.*, 2022). Triformer (Cirstea *et al.*, 2022) designs a patch attention with linear complexity and variable specific parameters to enhance accuracy. (Shabani *et al.*, 2022) develops a multi-scale framework to model time series using different resolutions and they utilize separate predictive models for each temporal scale which leads to high computational complexity. After the rise of generative pre-trained models, Large Language Models (LLMs) have been utilized for time series forecasting by fine-tuning and exploiting prompt engineering, e.g., Time-LLM (Jin *et al.*, 2023) which keeps the backbone LLM intact and input time series is reprogrammed with text prototypes before feeding it to the frozen LLM, aligning two modalities. GPT4TS (Zhou *et al.*, 2023) employs GPT-2 (Radford *et al.*, 2019) model for time series forecasting by feeding the time series patches to the model in a Channel-Independent manner. A spatial-temporal large language model is proposed for traffic forecasting (Liu *et al.*, 2024) by defining spatial-temporal embedding to learn the spatial locations and global temporal dependencies of time steps at each location. In traffic prediction often capturing spatial-temporal dependencies at multiple scales is required. To address the mentioned requirement, MT-STNets is designed in (Wang *et al.*, 2021b), for prediction of both fine-grained traffic conditions on individual roads and coarse-grained traffic flows across urban areas. More recently, Pathformer (Chen *et al.*, 2024) is proposed which exploits adaptive pathways to capture multi-scale temporal relations in an adaptive manner by automatically selecting patches of different resolutions, which uses separate set of parameters for each temporal granularity in its design. Different from the above mentioned models, which either disregard multi-scale modeling entirely or represent multi-scale information inside their model architecture, we devise a way to project multiple-scale temporal representations to the model dimension in the time series embedding phase and utilize the same model parameters to capture multi-scale and cross-scale information efficiently.

We expect improvements by using multi-scale embedding, since real-world time series often exhibit multiple seasonal patterns and modeling all the scales would improve the model performance. We employed Fourier analysis to calculate the high frequency components and dominant periods of time series samples from various benchmark datasets in order to show that real-world datasets usually rely on more than one seasonality pattern (scale). For example, electricity dataset typically exhibits daily (e.g., 24-hour or 48-hour) seasonality, together with long-term periods, such as monthly consumption or seasonal peaks (with periods of 90, 102). Multi-scale embedding and sharing the same model space among different scales is the main difference between our proposed method and the existing literature which exploits the interactions among different scales. We also use channel dimensionality reduction through 1-dimensional convolution over the channel dimension in the cross-channel encoder layer, which helps to reduce the noise effect in the highly correlated time series by mitigating the over-fitting chance. The previous methods usually utilize a single linear layer to map the model dimension to the prediction window, which would lead to over-fitting in longer horizon forecasting scenarios. Therefore, we design a simple but effective way to avoid this effect,

by decoding the extracted information through linear layers over consecutive steps.

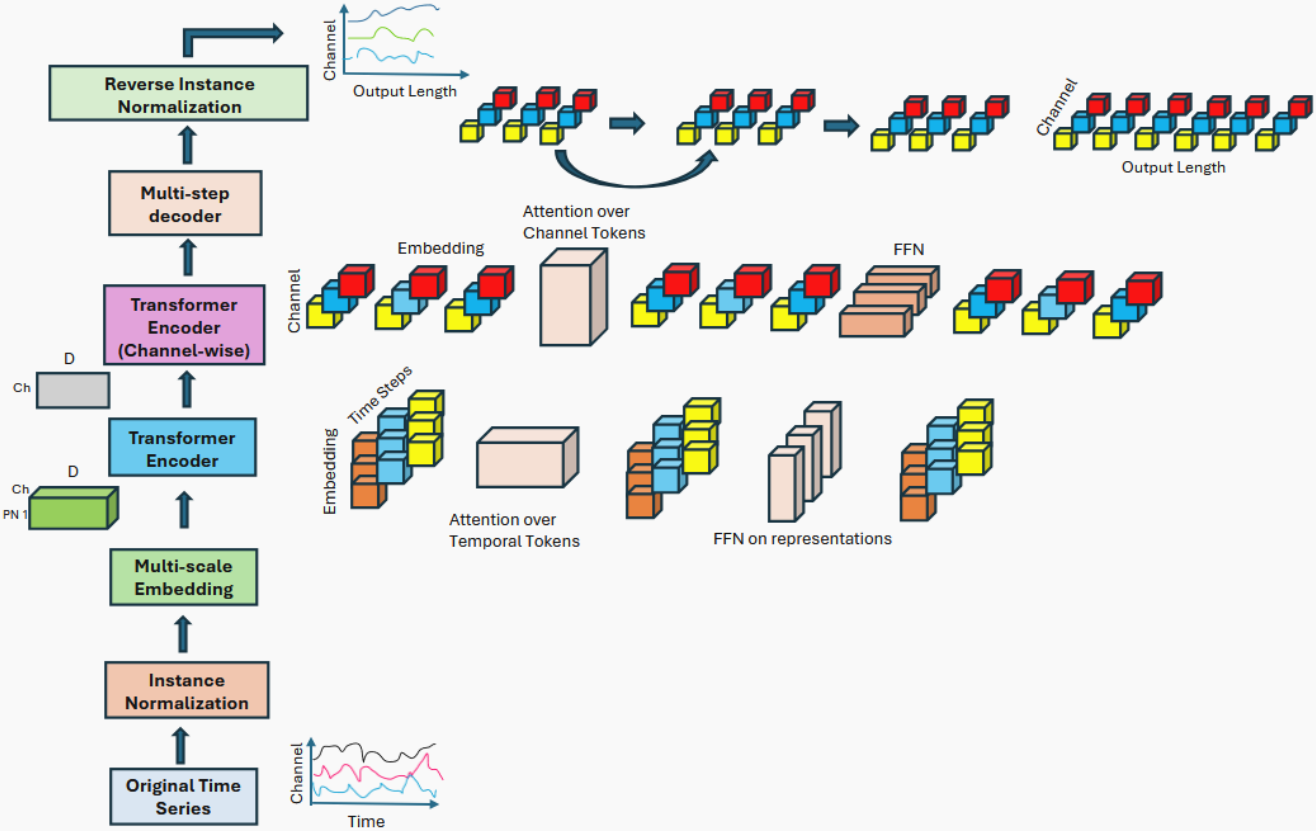


Figure 6.2 – General overview of the proposed framework (MultiPatchFormer). The main steps include normalization, multi-scale embedding using different patch sizes, temporal encoder, reshaping the output of the temporal encoder using a 1-dimensional convolution layer and sending the result to the channel-wise encoder. The output of the channel-wise encoder is passed through the multi-step decoder to generate the final predictions followed by the de-normalization step.

6.3 Methods

In this paper, we deal with the problem of time series forecasting which can be stated as following : Given a set of historical time series of C variates over L timestamps $(X_1, X_2, ..., X_L)$, with X_i indicating the observation at timestamp i of dimension C, we aim to predict F steps of the future time steps : $(X_{L+1}, ..., X_{L+F})$. To effectively capture multi-scale information and inter-series correlations, we propose MultiPatchFormer, which utilizes a novel predictor composed of a sequence of linear layers to simulate auto-regressive decoding in patch level. The overall architecture is illustrated in Figure 6.2 and Figure 6.3 demonstrates the multi-scale embedding layer in detail. The whole framework consists of instance normalization (Kim *et al.*, 2021), multi-scale embedding, temporal encoder, inter-series encoder module and predictor. Instance normalization (Kim *et al.*, 2021) is a technique employed by many models to avoid distribution shifts between training and test sets. The multi-scale embedding first divides the input series into patches of various sizes and then embeds all of them to the same model space. In other words, given 4 tem-

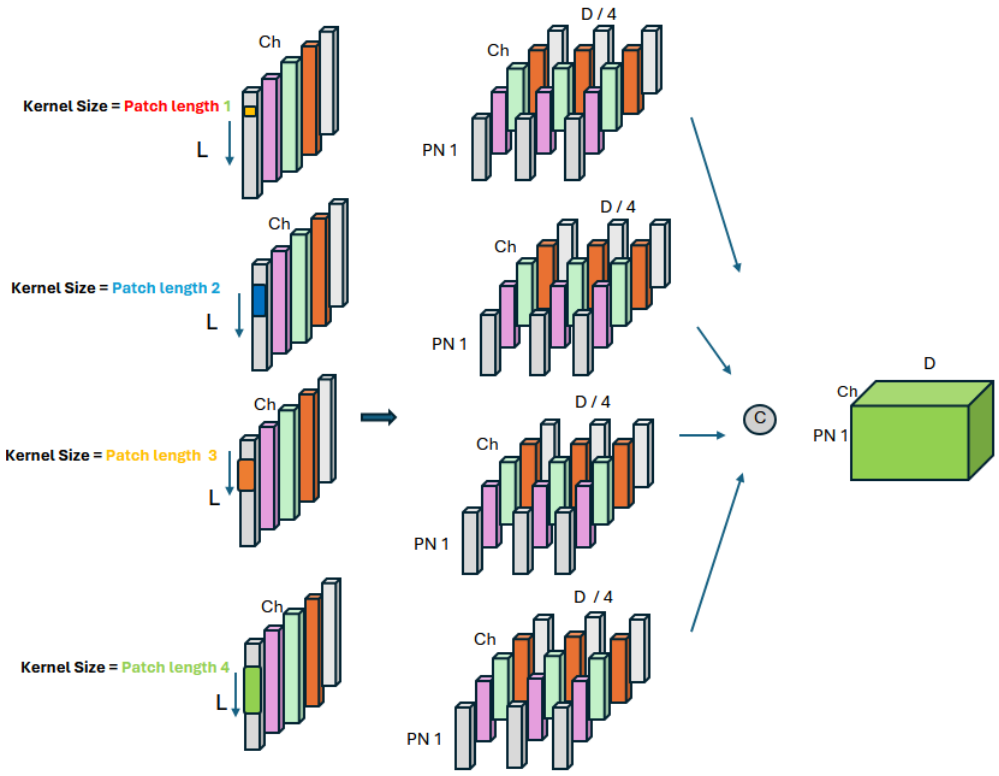


Figure 6.3 – Multi-scale embedding is illustrated using 4 different patch sizes (scales), where D refers to the model dimension (d_{model}), Ch indicates time series channels (variates), and $PN1$ refers to the number of patches fixed across four scales by choosing appropriate strides.

poral scales, time series is reshaped to $((B, C), L, 1)$, then is convolved with filters of different sizes and strides, corresponding to the patch sizes and strides (overlapping region between to consecutive patches). B, C and L indicate batch size, channels and series length, respectively. We deliberately decided to exploit one-dimensional convolution to split series for the purpose of capturing local temporal dynamics inside a patch (intra-patch relations). After splitting the input time series by using patches of four various lengths, they are merged together into the model dimension (d_{model}). To be more specific, the output channel of the individual convolutions are set to $\frac{d_{model}}{4}$, mapping each patched series into its own smaller sub-space and then merging them to the main model dimension. This strategy employs separate 1-dimensional convolutions with input channel of 1 and output channel of $\frac{d_{model}}{4}$ to project multi-scale information into the embedded space and each channel of time series is embedded independently in this manner, as we reshape the series into $((B, C), L, 1)$ before applying the convolutions. In order to reduce the over-fitting effect in channel-wise multi-head self-attention, we devise a simple but yet effective solution for reducing noise and time complexity of time series with large number of channels. The regular linear predictor which is usually used as the final layer of time-series forecasting models, is susceptible to over-fitting, specifically when the forecast horizon is relatively long. Rather than using a simple linear layer to map from model dimension to the prediction length, we break the final linear layer to multiple linear layers to decode the predictions in multiple steps by generating part of the prediction horizon at each step. In the following section, each component of our model is described in more detail.

6.3.1 Multi-scale Embedding

In Computer Vision, prior to the main backbone, RGB channels are embedded into a D-dimensional vector at every pixel. This embedding process serves to mix information from channels through the embedding layer. However, employing a similar variable-mixing embedding, such as embedding M variables into a D-dimensional vector per time step, proved to be unsuitable for time series data because of two primary reasons. Firstly, the difference between variables within a time series is far greater than that among RGB channels within an image (Cirstea *et al.*, 2022). A mere embedding layer is insufficient for capturing the intricate correlations among variables, which might lead to the loss of their individual behavior. We refer X_{in} as the C variables input time series of length L which is further divided into PN patches of patch length PL after proper padding (padding by repeating the last time element stride times). The stride S is specified as the length of non overlapping region between two consecutive patches. Then, the patches will be embedded into embedding vectors of dimension $\frac{d_{model}}{num_{scales}}$:

$$X_{emb} = Embed_s(X_{in}) \tag{6.1}$$

In which, $X_{emb} \in C \times PN \times \frac{d_{model}}{num_{scales}}$ is the input embedding for one temporal scale and $Embed_s$ denotes embedding with scale or patch length of s. As we mentioned earlier we conduct this patchify embedding in a fully-convolution way to simplify computation and capture local dynamics in various scales. First, we extend (unsqueeze) the shape to $X_{emb} \in C \times L \times 1$, and then we feed the padded X_{embed} into a 1-dimensional convolution layer with kernel size PL and stride S, which maps the input with one channel into $\frac{d_{model}}{num_{scales}}$ output channels. It should be mentioned that in above process, each of the C univariate time series is embedded independently which keeps the variate dimension intact and helps to model inter-variable dependencies in the following blocks. We repeat the above embedding process for num_{scales} times, each time with different patch size and stride to capture and embed different scales of the input series. We utilize appropriate stride sizes with each patch length to obtain the same number of patches (patch numbers), because the resultant embeddings are concatenated as the final representation. Therefore, The multi-scale embedding is the result of concatenating embeddings with various patch sizes which are combined along their model dimension :

$$X_{emb} = Concatenate(Embed_1(X_{in}), ..., Embed_{num_{scale}}(X_{in})) \tag{6.2}$$

In our model, the embeddings of multiple scales are projected into a shared feature space. For example, given 4 different patch sizes, we project the time series into the same feature space by dividing the model dimension into 4 segments and assigning the embedding result of each scale to one segment of the model dimension. In other words, each scale is first projected into $model_{dim}/4$ space and then these embeddings are concatenated along the model dimension. This allows for interaction between different temporal resolutions within the same representation space and capturing multi-scale and cross-scale information through the self-attention component.

6.3.2 Temporal Encoder

We define a set of different patch sizes $(P_1, P_2, ..., P_M)$ to represent various views corresponding to temporal resolution of input series. Each divided patch is embedded to a lower dimension than the main model dimension (to $\frac{d_{model}}{4}$ in case of using 4 different scales), which are then concatenated to produce the final embedding ready to be fed into the temporal and channel-wise encoders, respectively. The embedded series is first passed through the temporal encoder, which consists of a multi-head self attention followed by layer normalization and feed-forward layers to capture long-term temporal dependencies across multiple scales, simultaneously. The multi-head self-attention in temporal encoder, extracts temporal dependencies over the patches. Consider a set of divided patches which are the result of multiple scale division : $(P_1, P_2, ..., P_{PN})$, with PN indicating number of patches fixed for all scales and $P_i \in R^{d_{model}}$. Then, we employ trainable linear transformations to obtain query, key and value matrices, $Q, K, V \in R^{PN \times d}$ and perform attention score calculation as :

$$AttentionScore = softmax\left(\frac{Q \cdot K^T}{\sqrt{d}}\right) \tag{6.3}$$

In the above formula, d indicates the head dimension $\frac{d_{model}}{h}$, with h showing the number of heads. The attention scores are multiplied by the value matrix to obtain the result of multi-head self-attention over divided patches which demonstrates temporal dependencies along the patches.

6.3.3 Channel-wise Attention

The output of the temporal encoder is passed to the channel-wise encoder, which employs a cross-channel multi-head attention module to capture the correlations among the channels (variates). However, the shape of resultant tensor of temporal encoder is $(B \times C) \times PN \times d_{model}$, which is reshaped into $B \times C \times (PN \times d_{model})$. Then we exploit a 1-dimensional convolution to reduce the last dimension into d_{model} (model dimension), preparing it to be fed into the channel-wise encoder layer. Therefore, the input of the channel-wise multi-head attention is of shape $B \times C \times d_{model}$. When performing attention over channel dimension, the vanilla self-attention may suffer from high computation and over-fitting, specially when dealing with a high-dimensional dataset with many variates. To reduce the noise over-fitting issue and computational complexity, we devise a solution to summarize the key and value matrices in channel-wise attention step. In other words, query matrix remains intact of shape $B \times C \times d$ and we apply a 1-dimensional convolution along the channel dimension with the same kernel size and stride (with proper padding) to reduce the number of channels in key and value matrices, resulting in tensors of size $B \times r \times d$ with $r < C$ which indicates the summarized channels. We deploy kernel and stride size according to the dataset in out experiments, e.g. kernel size and stride of 21 with padding 10 on each side of the input series are utilized for Traffic dataset. Therefore, it leads to decrease in time complexity and noise effect when calculating attention scores over the channel dimension :

$$AttentionScore = softmax\left(\frac{Q_{channel} \cdot K_{channel}^T}{\sqrt{d}}\right) \tag{6.4}$$

In which $Q_{channel} \cdot K^T_{channel}$ requires $C \times d \times r$ multiplications rather than $C^2 \times d$. The resultant attention output is further passed through layer normalization and feed-forward layers to extract meaningful features considering the dependencies over the channel dimension.

6.3.4 Multi-step Decoder

Both long-term temporal and channel-wise dependencies are captured through the employed encoders in previous steps. The final output which comes from the encoder over channel dimension is of dimension $B \times C \times d_{model}$. Typically, a linear layer is employed as the final decoder or predictor to map from model dimension to prediction length (forecast window). But, the number of parameters in the output linear layer increases when dealing with long prediction horizons which might lead to over-fitting and affecting the far-future predictions by noise. To mitigate this issue, we propose a multi-step decoding process in which a sequence of linear layers are applied to the encoded feature map from the channel-wise encoder and current prediction results. In other words, we first divide the prediction length into different parts (length of 4 or 8) and generate the prediction part by part, by applying the combination of the final feature map and previously predicted parts to a linear layer. The multi-step decoding process is illustrated in Figure 6.4.

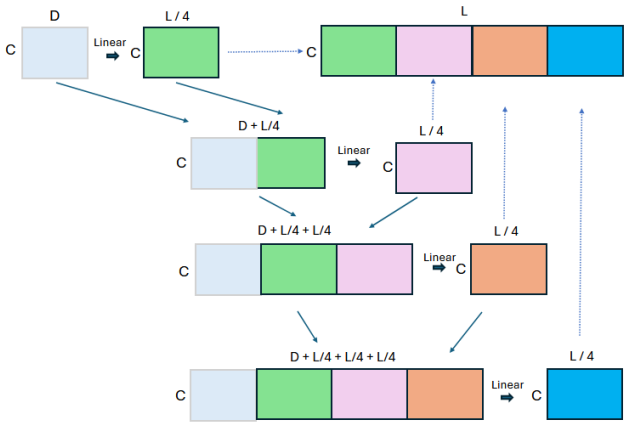


Figure 6.4 – Multi-step decoder is proposed to reduce noise effect in long prediction length. C, D and L refer to channels, model dimension and prediction length, respectively. At each step the feature map from the channel-wise encoder is concatenated with previously generated prediction segments and passed through a linear layer to generate the next prediction segment (part). The final prediction is the concatenation result of the individual predicted segments.

6.4 Results

6.4.1 Datasets and Baselines

Datasets We conduct different experiments using 7 real world datasets including electricity Transformer Temperature (Wu *et al.*, 2021), weather forecasting (Wu *et al.*, 2021), traffic (Wu *et al.*, 2021) and electricity consumption (Wu *et al.*, 2021). These datasets consists of the well-known ETT (ETTh1, ETTh2, ETTm1, ETTm2), Weather, Electricity and Traffic. More details about the datasets are reported in Table 6.1. **Baseline and**

metrics 14 well-known time series forecasting models are selected as baselines, including Pathformer (Chen *et al.*, 2024), PatchTST (Nie *et al.*, 2022), DLinear (Zeng *et al.*, 2023), NLinear (Zeng *et al.*, 2023), Crossformer (Zhang et Yan, 2022), Scaleformer (Shabani *et al.*, 2022), TiDE (Das *et al.*, 2023), FEDformer (Zhou *et al.*, 2022), Pyraformer (Liu *et al.*, 2021b), Autoformer (Wu *et al.*, 2021), Time-LLM (Jin *et al.*, 2023), GPT4TS (Zhou *et al.*, 2023), MTGNN (Wu *et al.*, 2020b) and SDGL (Li *et al.*, 2023b). To ensure fairness in our experiments, the input length is fixed for all models (L = 96) and prediction length consists of (F = 96, 192, 336, 720). Since the authors of Time-LLM used input length of 512 in their paper (Jin *et al.*, 2023), we set the context window to 512 in LLM comparison experiments and trained our model on different datasets with input length of 512 and prediction length in {96, 192, 336, 512}. We compare our model with graph models based on input window of length 96. Two well-known error metrics in time series forecasting are selected : Mean Absolute Error (MAE) and Mean Square Error (MSE) to compare the models. **Implementation details** We utilize Adam optimizer with learning rate in $\{10^{-3}, 10^{-4}\}$ and L1 loss function. The models are trained for 10 epochs with early stopping (patience of 3) based on the validation loss. We implemented our model using Pytorch framework (Paszke *et al.*, 2019) and the experiments are executed on an NVIDIA A100 40GB GPU. MultiPatchFormer utilizes 4 different scales to patchify and embed input time series (common scales in the time series forecasting domain, e.g., 8, 16, 24, 48 with proper strides, 8, 8, 7, 6 to create the same number of patches).

Table 6.1 – Detailed dataset descriptions. Dim indicates the variate number of each dataset. Dataset Size denotes the total number of time points in (Train, Validation, Test) splits, respectively. Mean and maximum cross-correlations (Dynamic Time Warping based) between the channels of each dataset are also reported in Mean Corr and Max Corr columns. Trend column indicates strength of trend for each dataset.

Dataset	Dim	Pred Length	Dataset Size	Mean Corr	Max Corr	Trend	Freq	Field
Electricity	321	{96, 192, 336, 720}	(18317, 2633, 5261)	0.981	0.999	0.315	Hourly	Electricity
ETTh1, ETTh2	7	{96, 192, 336, 720}	(8545, 2881, 2881)	0.551 (0.551)	0.897 (0.946)	0.797 (0.761)	Hourly	Electricity
ETTm1, ETTm2	7	{96, 192, 336, 720}	(34465, 11521, 11521)	0.551 (0.551)	0.897 (0.945)	0.896 (0.822)	15 min	Electricity
Weather	21	{96, 192, 336, 720}	(36792, 5271, 10540)	0.704	0.999	0.039	10 min	Weather
Traffic	862	{96, 192, 336, 720}	(12185, 1757, 3509)	0.751	0.985	0.031	Hourly	Transportation

6.4.2 Main Results

The main results are summarized in Tables 6.2, 6.3 and 6.4. The results related to the baseline models are taken from (Chen *et al.*, 2024) and we follow the same settings to train and evaluate our model. All experiments are repeated 5 times and the average results are reported. According to Table 6.2, our MultiPatchFormer, shows better performance across four different prediction horizons. Regarding error measures across four prediction lengths, our model achieves the best or second best performance among the state-of-the-art models in 82% of cases in MSE metric and 86% of the conducted experiments in terms of MAE measure. Our MultiPatchFormer outperforms the baseline models on benchmarks with a high number of variates and complex structure. For instance, in Traffic dataset (862 covariates), MultiPatchFormer persistently outperforms the second best baseline by more than 5% on average MSE and 7.7% on average MAE across four prediction windows, while consuming less training time and parameters. By exploiting 321 variates in ECL (Electricity) dataset, we achieved average error reduction of 3.7% compared to Pathformer (Chen *et al.*, 2024) and 10.7% improvement over the PatchTST model. This highlights that MultiPatchFor-

mer is capable of utilizing extensive covariate dependencies for high accuracy prediction. Remarkably, our model makes predictions with lower error in complex problems, including Electricity (ECL) and Traffic, which underscores the efficiency of the channel-wise attention and multi-scale embedding to capture the inter-series dependencies across various scales. It worth noting that while NLinear (Zeng *et al.*, 2023) achieves promising performance in some datasets, MultiPatchFormer significantly outperforms the MLP-based models (NLinear and DLinear) by over 26.8% error reduction on Traffic and more than 10% improvement on Electricity dataset. This indicates that Transformer models are still powerful in forecasting long-term time series tasks. In terms of cross-variate modeling, Crossformer (Zhang et Yan, 2022), utilizes a variant of attention to capture those type of correlations, but our model gains impressive improvement of over 28% and 18% compared to the Crossformer model on the Electricity and Traffic datasets, respectively. According to Table 6.3, our model consistently outperforms or achieves similar performance to LLM-based methods while using significantly fewer parameters. Notably, compared to Time-LLM, which utilizes 7 billion parameters, our model maintains competitive performance across different datasets. For example, when time series forecasting of the ETTh1 dataset at a prediction window of 720, MultiPatchFormer obtains an MSE of 0.434, lower than Time-LLM's 0.442 while consuming far fewer computational resources. This underscores the effectiveness of our approach relative to larger models. Graph learning models have been proved to be promising in forecasting traffic flow by modeling the temporal correlations and the spatial dependencies between the variables through the graph learning strategy (Wu *et al.*, 2020b) and (Li *et al.*, 2023b). We compare our MultiPatchFormer with the spatio-temporal graph models on various benchmarks, specially on Traffic dataset. As illustrated by Table 6.4, our model outperforms SDGL and MTGNN with a large margin, particularly on Traffic forecasting task, with average MSE improvement of 23% and 28%, respectively.

As shown in Table 6.2, MultiPatchFormer consistently achieves low error rates for datasets with strong seasonality and highly correlated datasets with many variates (Electricity, Traffic, Weather). This can be attributed to the multi-scale embedding and temporal Transformer blocks, which effectively captures temporal correlations at various granularities and channel-wise attention, which represents the complex dependencies between time series channels through learning of the attention scores between channel pairs, extracting meaningful features. However, our model exhibits relatively lower performance on datasets with dominating short-term dependencies and limited training data, such as the ETTh2 dataset. This could be associated with the multi-scale embedding and multi-head attention, which, while being effective for capturing long-range patterns, may not show significant effectiveness for data with simpler temporal patterns and limited training samples.

In time series forecasting, size of the input series determines the historical information that a model receives to forecast. Different input lengths are configured to verify effectiveness of MultiPatchFormer to deal with various input lengths. As the input length increases, the error measures of MultiPatchFormer continue to decrease, demonstrating its robustness in handling long and noisy sequences evidenced by Table 6.5, which indicates an MSE reduction of more than 15%, 11% and 10% achieved on the ETTm1, Weather and Electricity datasets when prolonging the input length from 96 to 720. In this experiment, the best performing predictive models are selected and the results for input length of 48, 192 and 336 are visualized in Figure 6.5. According to this figure, MultiPatchFormer consistently outperforms other baselines in most cases by using longer input lengths. This serves as evidence of efficiency of MultiPatchFormer in utilizing information of long input sequences thanks to multi-scale analysis and capturing cross-variable dependencies. MultiPatchFormer is also compared with other best performing baselines using a shorter input length (e.g. 48) on Electricity dataset. As Figure 6.5 illustrates, our model forecasts different prediction lengths by using a short input window (H = 48) with the lowest error, indicating its capability in capturing temporal correlations from shorter input sequences.

Table 6.2 – Multivariate time series forecasting results. The input length (lookback window) is fixed to 96, and prediction length is in {96, 192, 336, 720}.

Models		Ours		Pathformer		PatchTST		DLinear		NLinear		Crossformer		Scaleformer		TiDE		FEDformer		Autoformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	**0.315**	**0.342**	*0.316*	*0.346*	0.324	0.361	0.392	0.405	0.386	0.392	0.429	0.440	0.396	0.440	0.356	0.381	0.379	0.419	0.505	0.475
	192	0.367	**0.369**	*0.366*	*0.370*	**0.362**	0.383	0.441	0.436	0.440	0.430	0.494	0.482	0.434	0.460	0.391	0.399	0.426	0.441	0.553	0.496
	336	0.399	**0.394**	**0.386**	**0.394**	*0.390*	*0.402*	0.501	0.478	0.480	0.443	0.706	0.625	0.462	0.476	0.424	0.423	0.445	0.459	0.621	0.537
	720	0.464	**0.432**	**0.460**	**0.432**	*0.461*	*0.438*	0.538	0.526	0.486	0.472	0.750	0.689	0.494	0.500	0.480	0.465	0.543	0.490	0.671	0.561
	Avg	0.386	**0.384**	**0.383**	*0.386*	*0.384*	0.396	0.468	0.461	0.448	0.434	0.595	0.559	0.447	0.469	0.413	0.417	0.448	0.452	0.588	0.517
ETTm2	96	*0.171*	*0.250*	**0.170**	**0.248**	0.177	0.260	0.186	0.279	0.177	0.257	0.197	0.321	0.182	0.275	0.182	0.264	0.203	0.287	0.255	0.339
	192	**0.236**	**0.294**	*0.238*	*0.295*	0.248	0.306	0.266	0.339	0.241	0.297	0.326	0.375	0.251	0.318	0.256	0.323	0.269	0.328	0.281	0.340
	336	*0.303*	*0.337*	**0.293**	**0.331**	0.304	0.342	0.332	0.376	0.302	0.337	0.372	0.421	0.340	0.375	0.313	0.354	0.325	0.366	0.339	0.372
	720	*0.397*	0.393	**0.390**	**0.389**	0.403	0.397	0.462	0.455	0.405	*0.396*	0.410	0.448	0.435	0.433	0.419	0.410	0.421	0.415	0.433	0.432
	Avg	*0.277*	*0.319*	**0.273**	**0.316**	0.283	0.326	0.312	0.362	0.281	0.322	0.326	0.391	0.302	0.350	0.293	0.338	0.305	0.349	0.327	0.371
ETTh1	96	*0.378*	**0.389**	0.382	0.400	0.394	0.408	0.392	0.405	0.386	0.392	0.429	0.440	0.396	0.440	0.427	0.450	**0.376**	0.419	0.449	0.459
	192	**0.430**	**0.420**	0.440	0.427	0.446	0.438	0.441	0.436	0.440	0.430	0.494	0.482	0.434	0.460	0.472	0.486	0.420	0.448	0.500	0.482
	336	*0.473*	0.442	**0.454**	**0.432**	0.485	0.455	0.501	0.478	0.480	*0.443*	0.706	0.689	0.462	0.476	0.527	0.527	0.459	0.465	0.521	0.496
	720	**0.475**	*0.466*	*0.479*	**0.461**	0.495	0.474	0.538	0.526	0.486	0.472	0.750	0.689	0.494	0.500	0.644	0.605	0.506	0.507	0.514	0.512
	Avg	**0.439**	**0.429**	**0.439**	*0.430*	0.455	0.444	0.468	0.461	0.448	0.434	0.595	0.575	*0.447*	0.469	0.518	0.517	0.440	0.460	0.496	0.487
ETTh2	96	*0.287*	*0.333*	**0.279**	**0.331**	0.294	0.343	0.331	0.381	0.290	0.339	0.632	0.547	0.364	0.407	0.304	0.359	0.346	0.388	0.358	0.397
	192	*0.366*	*0.383*	**0.349**	**0.380**	0.378	0.394	0.432	0.435	0.379	0.395	0.876	0.663	0.466	0.458	0.394	0.422	0.429	0.439	0.456	0.452
	336	0.413	0.420	**0.348**	**0.382**	*0.382*	*0.410*	0.441	0.451	0.421	0.431	0.924	0.702	0.479	0.476	0.385	0.421	0.496	0.487	0.482	0.486
	720	0.417	0.435	**0.398**	**0.424**	*0.412*	*0.433*	0.564	0.578	0.436	0.453	1.390	0.863	0.487	0.492	0.463	0.475	0.463	0.474	0.515	0.511
	Avg	0.372	*0.394*	**0.344**	**0.379**	*0.367*	0.395	0.442	0.461	0.382	0.405	0.956	0.694	0.449	0.458	0.387	0.419	0.434	0.447	0.453	0.462
Weather	96	*0.157*	*0.197*	**0.156**	**0.192**	0.177	0.218	0.195	0.253	0.168	0.208	0.181	0.231	0.288	0.365	0.202	0.261	0.238	0.314	0.249	0.329
	192	*0.207*	*0.242*	**0.206**	**0.240**	0.224	0.258	0.239	0.299	0.217	0.255	0.219	0.275	0.368	0.425	0.242	0.298	0.275	0.329	0.325	0.370
	336	*0.267*	*0.286*	**0.254**	**0.282**	0.277	0.297	0.282	0.333	0.267	0.292	0.274	0.332	0.447	0.469	0.287	0.335	0.339	0.377	0.351	0.391
	720	*0.345*	*0.339*	**0.340**	**0.336**	0.350	0.345	0.352	0.390	0.351	0.346	0.356	0.387	0.640	0.574	0.351	0.386	0.389	0.409	0.415	0.426
	Avg	*0.244*	*0.266*	**0.239**	**0.263**	0.257	0.280	0.267	0.319	0.251	0.275	0.258	0.306	0.436	0.458	0.271	0.320	0.310	0.357	0.335	0.379
Electricity	96	*0.146*	**0.233**	**0.145**	*0.236*	0.180	0.264	0.194	0.276	0.185	0.266	0.254	0.347	0.182	0.297	0.194	0.277	0.186	0.302	0.196	0.313
	192	**0.163**	**0.247**	*0.167*	*0.256*	0.188	0.275	0.193	0.279	0.189	0.276	0.261	0.353	0.188	0.300	0.193	0.280	0.197	0.311	0.211	0.324
	336	**0.178**	**0.263**	*0.186*	*0.275*	0.206	0.291	0.206	0.294	0.204	0.289	0.273	0.364	0.210	0.324	0.206	0.296	0.213	0.328	0.214	0.327
	720	**0.213**	**0.293**	*0.231*	*0.309*	0.247	0.328	0.241	0.328	0.245	0.319	0.303	0.388	0.232	0.339	0.242	0.328	0.233	0.344	0.236	0.342
	Avg	**0.175**	**0.259**	*0.182*	*0.269*	0.205	0.290	0.209	0.294	0.206	0.288	0.273	0.363	0.203	0.315	0.209	0.295	0.207	0.321	0.214	0.327
Traffic	96	**0.438**	**0.260**	*0.479*	*0.283*	0.492	0.324	0.648	0.396	0.645	0.388	0.558	0.320	2.678	1.071	0.568	0.352	0.576	0.359	0.597	0.371
	192	**0.456**	**0.268**	*0.484*	*0.292*	0.487	0.303	0.613	0.386	0.599	0.365	0.572	0.331	0.564	0.351	0.612	0.371	0.610	0.380	0.607	0.382
	336	**0.475**	**0.276**	*0.503*	*0.299*	0.505	0.317	0.614	0.383	0.606	0.367	0.587	0.342	0.570	0.349	0.605	0.374	0.608	0.375	0.623	0.387
	512	**0.514**	**0.295**	*0.537*	*0.322*	0.542	0.337	0.655	0.405	0.645	0.388	0.652	0.359	0.576	0.349	0.647	0.410	0.621	0.375	0.639	0.395
	Avg	**0.470**	**0.275**	*0.501*	*0.299*	0.507	0.320	0.632	0.393	0.624	0.377	0.592	0.338	1.097	0.530	0.608	0.377	0.604	0.372	0.617	0.384

Table 6.3 – Comparison result of our model with LLM models. The input length (lookback window) is fixed to 512 and prediction length is in {96, 192, 336, 720}.

Methods		Ours		Time-LLM		GPT4TS	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
Traffic	96	0.370	**0.236**	**0.362**	0.248	0.388	0.282
	192	0.383	**0.242**	**0.374**	0.247	0.407	0.290
	336	0.398	**0.250**	**0.385**	0.271	0.412	0.294
	720	0.433	**0.270**	**0.430**	0.288	0.450	0.312
	Avg	0.396	**0.250**	**0.388**	0.264	0.414	0.294
Electricity	96	**0.131**	**0.222**	0.131	0.224	0.139	0.238
	192	**0.149**	**0.239**	0.152	0.241	0.153	0.251
	336	0.162	0.253	**0.160**	**0.248**	0.169	0.266
	720	**0.191**	**0.277**	0.192	0.298	0.206	0.297
	Avg	**0.158**	**0.248**	**0.158**	0.252	0.167	0.263
ETTh1	96	0.366	**0.392**	**0.362**	**0.392**	0.376	0.397
	192	0.400	**0.418**	**0.398**	**0.418**	0.416	0.418
	336	**0.423**	0.437	0.430	**0.427**	0.442	0.433
	720	**0.434**	0.459	0.442	**0.457**	0.477	0.456
	Avg	**0.406**	0.427	0.408	**0.423**	0.465	0.455
Weather	96	**0.144**	**0.185**	**0.147**	0.201	0.162	0.212
	192	0.190	**0.231**	**0.189**	0.234	0.204	0.248
	336	**0.242**	**0.270**	**0.262**	0.279	0.254	0.286
	720	0.313	0.323	**0.304**	**0.316**	0.326	0.337
	Avg	**0.222**	**0.252**	0.225	0.257	0.237	0.270

6.4.3 Ablation Studies

To verify the effectiveness of the components in our design of MultiPatchFormer, we conduct ablation studies by removing the multi-scale embedding, channel-wise encoder and multi-step decoder from the main model. Time series forecasting results using our model without those components are reported in Table 8.7. As evidenced by the table, each of the multi-scale embedding, channel-wise encoder and multi-step decoder modules contribute to performance promotion. For example, in ETTh1 forecasting dataset, multi-scale embedding improves the MSE error rate by approximately 2% in prediction length of 720 and the channel-wise encoder promotes the prediction accuracy (MSE) by 2.5%. Our multi-step decoder, improves the prediction error in most cases, specifically when the forecast horizon is long, e.g. 720. For example, in traffic forecasting, consisting of 862 variables across 720 future timestamps, the utilization of a multi-step decoder yields an MAE error reduction of 1%. We utilize a different kernel size for each dataset in the channel summarization part of the channel-wise attention in order to project the key and values, depending on

Table 6.4 – Comparison result of our model with graph models. The input length (lookback window) is set to 96 and prediction length is in {96, 192, 336, 720}.

Methods		Ours		SDGL		MTGNN	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
Traffic	96	**0.438**	**0.260**	0.557	0.271	0.660	0.437
	192	**0.456**	**0.268**	0.581	0.287	0.649	0.438
	336	**0.475**	**0.276**	0.611	0.302	0.653	0.472
	720	**0.514**	**0.295**	0.702	0.345	0.639	0.437
	Avg	**0.470**	**0.275**	0.613	0.301	0.650	0.446
Electricity	96	0.146	**0.233**	**0.145**	0.238	0.217	0.318
	192	0.163	**0.247**	**0.159**	0.255	0.238	0.352
	336	**0.178**	**0.263**	0.185	0.284	0.260	0.348
	720	**0.213**	**0.293**	0.232	0.314	0.290	0.369
	Avg	**0.175**	**0.259**	0.180	0.273	0.251	0.347
ETTh1	96	**0.378**	**0.389**	0.446	0.428	0.515	0.517
	192	**0.430**	**0.420**	0.471	0.452	0.553	0.522
	336	**0.473**	**0.442**	0.506	0.475	0.612	0.577
	720	**0.475**	**0.466**	0.623	0.548	0.609	0.597
	Avg	**0.439**	**0.429**	0.512	0.476	0.572	0.553
Weather	96	0.157	0.197	**0.153**	**0.194**	0.230	0.329
	192	**0.207**	**0.242**	0.212	0.257	0.263	0.322
	336	0.267	**0.286**	**0.263**	0.294	0.354	0.396
	720	**0.345**	**0.339**	**0.352**	0.360	0.409	0.371
	Avg	**0.244**	**0.266**	0.245	0.276	0.314	0.355

the performance improvement. In some cases, e.g., Electricity dataset, the kernel size is set to 1, since it gives the best results compared to the larger kernels. But, in datasets with highly correlated channels (such as Traffic with 862 variates), large kernels (e.g., 21) yield lower error rates by reducing channel dimension in key and value of the channel-wise attention. We study the impact of varying number of scales on time series forecasting and indicate the results in Table 6.7. In some cases, considering only one dominant scale is enough, but in some datasets, utilizing two or more than two scales helps to boost the performance and capture cross-scale information. In addition, more experiments are conducted in order to verify the effect of hyper-parameters, including the learning rate, hidden (model) dimension and number of training epoch. The results are illustrated in Figure 6.6.

Table 6.5 – Multivariate time series forecasting results of MultiPatchFormer (ours) with different input length {96, 192, 336, 512, 720} averaged over four prediction lengths.

Input Length	96		192		336		512		720	
Metric	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
ETTm1	0.396	0.389	0.365	0.379	0.352	0.372	**0.350**	**0.372**	0.353	0.376
ETTm2	0.279	0.321	0.263	0.315	0.253	**0.307**	**0.252**	0.309	0.256	0.313
Weather	0.249	0.270	0.232	0.259	0.224	0.254	**0.222**	**0.254**	0.224	0.257
ECL	0.175	0.259	0.163	0.250	0.161	0.249	0.159	0.249	**0.158**	**0.248**

6.4.4 Model Efficiency

Our model is versatile in terms of speed and memory usage efficiency. Specifically, MultiPatchFormer is comparable with other most recent models, including PatchTST (Nie *et al.*, 2022) and Pathformer (Chen *et al.*, 2024) in terms of training time and memory footprint. As illustrated in Figure 6.7, our model is faster than its main competitor model (Pathformer) in terms of training time, and its memory footprint is lower than most of the baselines, being close to memory usage of the linear models (DLinear). In time-series forecasting of ETTh1 dataset, with input length of 96 and prediction length of 96, our model is more than ten times faster than Pathformer and FEDformer, demonstrating lower memory consumption, while delivering the similar or better prediction accuracy. This highlights MultiPatchFormer's efficiency while providing accurate forecasts.

6.4.5 Visualization

We visualize prediction results of MultiPatchFormer for Electricity and Traffic datasets. As illustrated in Figure 6.8 and 6.9, the prediction curves align well with the ground truth ones in various cases and prediction horizons, indicating MultiPatchFormer's capability in handling complex trends and patterns.

6.5 Conclusion

We propose MultiPatchFormer, a Transformer based time series forecasting model, which integrates temporal dependencies associated with different temporal scales and captures intricate correlations among time series channels. In addition, we utilize 1-dimensional convolution to reduce channel dimension of key and value in the channel-wise multi-head attention to decrease noise effect and computational burden. A novel multi-step decoder is further devised to reduce over-fitting effect in dealing with long forecasting horizons. These innovative components collectively empower our model to produce accurate forecasts in variety of domains. Extensive experiments show the superior performance and robustness of our model in different settings and with varying input lengths.

Table 6.6 – Ablation of MultiPatchFormer model without different components : multi-scale embedding, channel-wise attention and multi-step decoder.

		W/O Multi-scale Embedding		**W/O Channel-wise encoder**		**W/O Multi-step decoder**		**MultiPatchFormer**	
Metric		**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
ETTh1	96	0.383	0.394	0.387	**0.393**	0.382	0.395	**0.380**	**0.393**
	192	0.435	**0.424**	0.443	0.425	0.435	0.427	**0.432**	**0.424**
	336	0.475	**0.444**	0.485	0.445	0.474	0.449	**0.472**	**0.444**
	720	0.478	0.470	0.481	**0.464**	0.473	0.471	**0.469**	**0.464**
Traffic	96	**0.437**	0.264	0.490	0.311	0.439	**0.260**	0.439	**0.260**
	192	0.460	0.271	0.493	0.306	**0.456**	**0.269**	0.457	**0.269**
	336	0.483	0.280	0.503	0.301	**0.477**	0.278	0.478	**0.277**
	720	0.516	0.300	0.537	0.319	0.516	0.299	**0.515**	**0.297**
Electricity	96	0.147	0.235	0.167	0.248	**0.146**	0.234	**0.146**	**0.233**
	192	0.164	0.248	0.181	0.259	**0.163**	**0.247**	**0.163**	**0.247**
	336	0.179	0.265	0.196	0.275	**0.178**	0.264	**0.178**	**0.263**
	720	0.216	0.295	0.235	0.308	0.216	0.295	**0.213**	**0.293**

Table 6.7 – Ablation study on the number of scales in multi-scale embedding.

Number of Scales		1		2		4	
Dataset		MAE	MSE	MAE	MSE	MAE	MSE
Electricity	96	0.138	0.229	**0.128**	**0.218**	0.131	0.222
	192	0.150	**0.239**	0.155	0.244	**0.149**	**0.239**
	336	0.165	0.255	0.168	0.258	**0.162**	**0.253**
	720	0.195	0.280	0.192	**0.276**	**0.191**	0.277
ETTm1	96	**0.309**	**0.338**	0.310	0.339	0.313	0.343
	192	**0.362**	**0.369**	0.368	0.370	0.365	**0.369**
	336	0.398	**0.393**	**0.396**	**0.393**	**0.396**	**0.393**
	720	0.472	0.434	**0.465**	**0.433**	0.470	0.434
ETTh1	96	0.380	0.390	0.379	**0.389**	**0.378**	**0.389**
	192	0.432	**0.420**	0.432	0.421	**0.430**	**0.420**
	336	0.477	0.444	0.474	0.442	**0.473**	**0.441**
	720	0.482	0.471	0.486	0.476	**0.476**	**0.468**

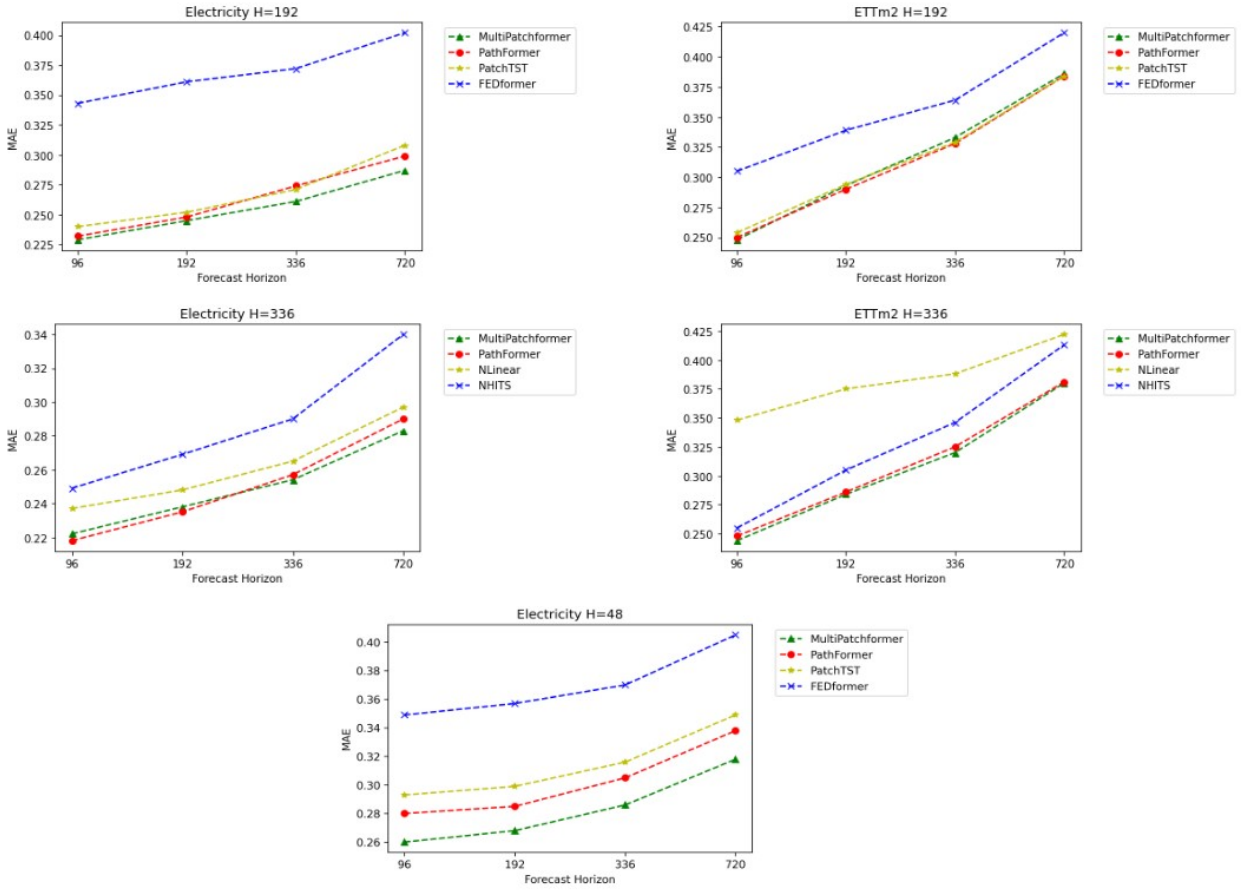


Figure 6.5 – Time series forecasting results of MultiPatchFormer with different input lengths (H = 48, 192, and 336) on ETTm2 and Electricity datasets.

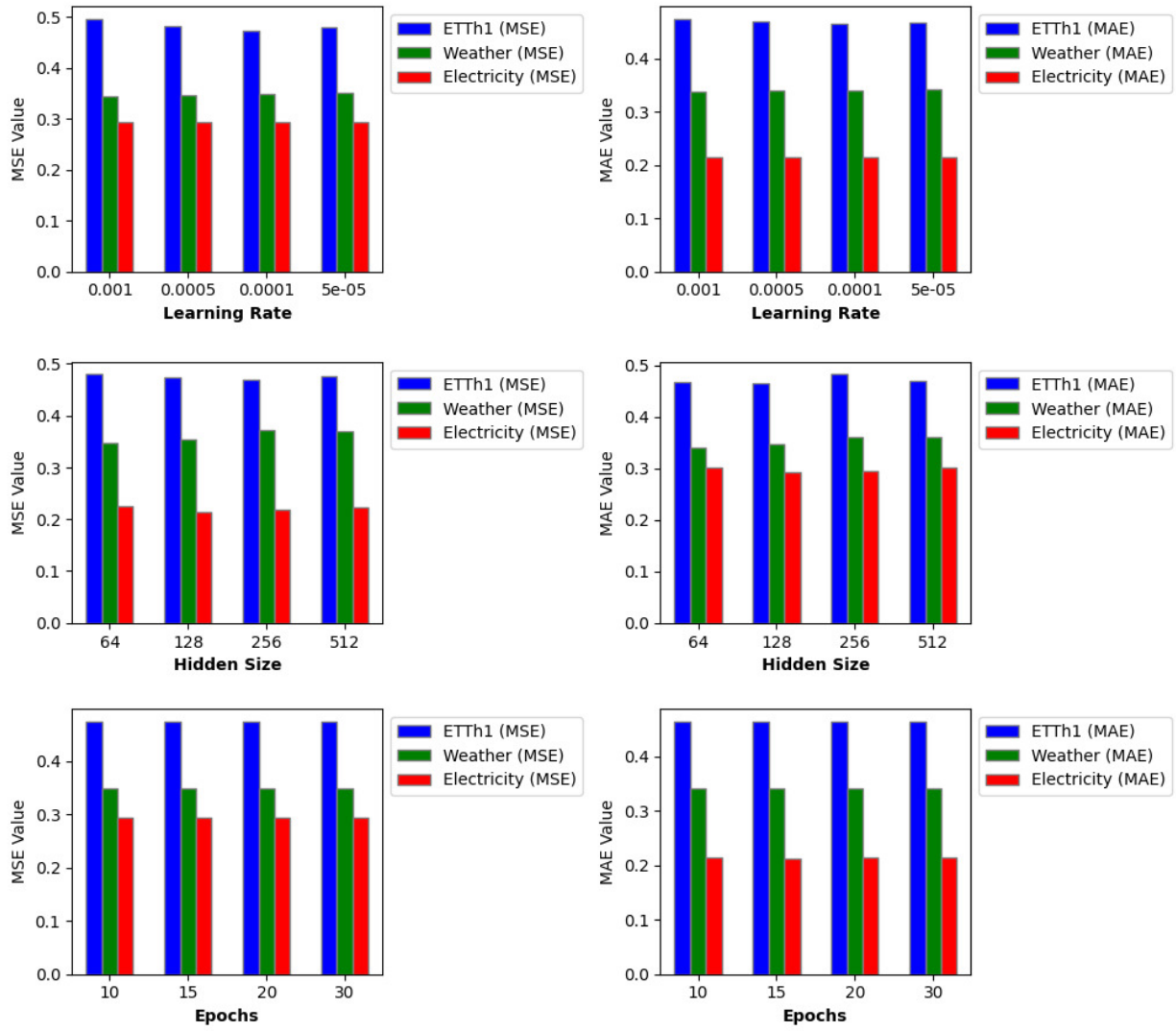


Figure 6.6 – Influence of hyper-parameters on time series forecasting of MultiPatchFormer.

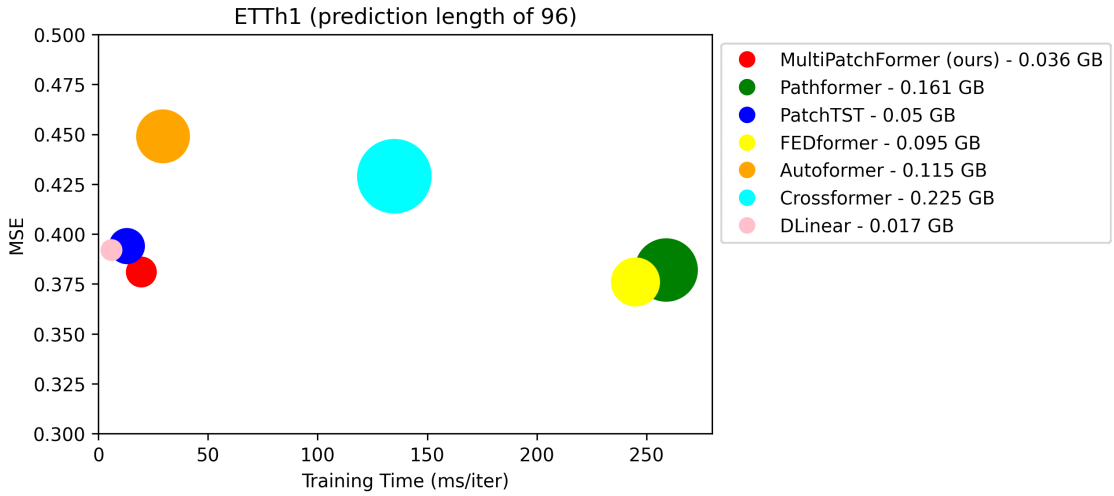


Figure 6.7 – Model efficiency comparison. Greater circle diameter represents larger memory footprint (in GB).

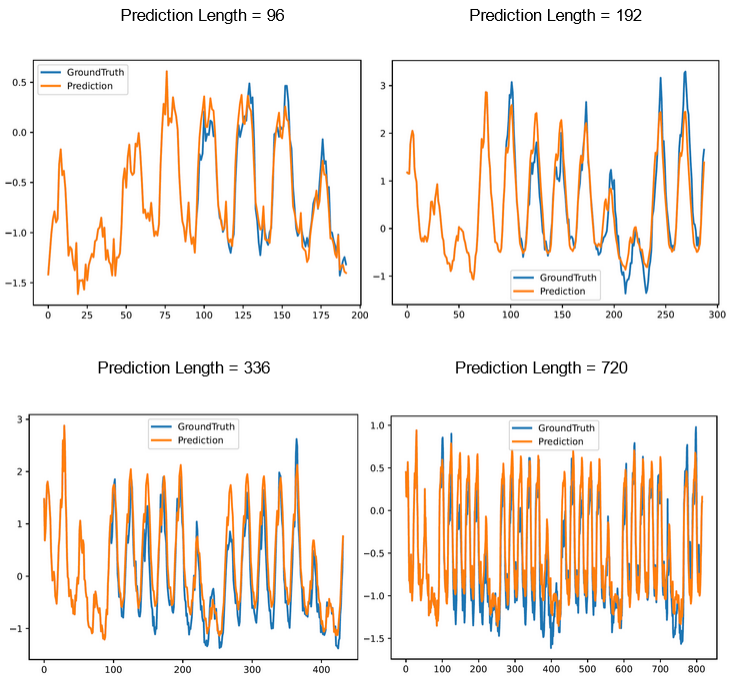


Figure 6.8 – Prediction results of our model applied to the Electricity dataset with different forecasting horizons.

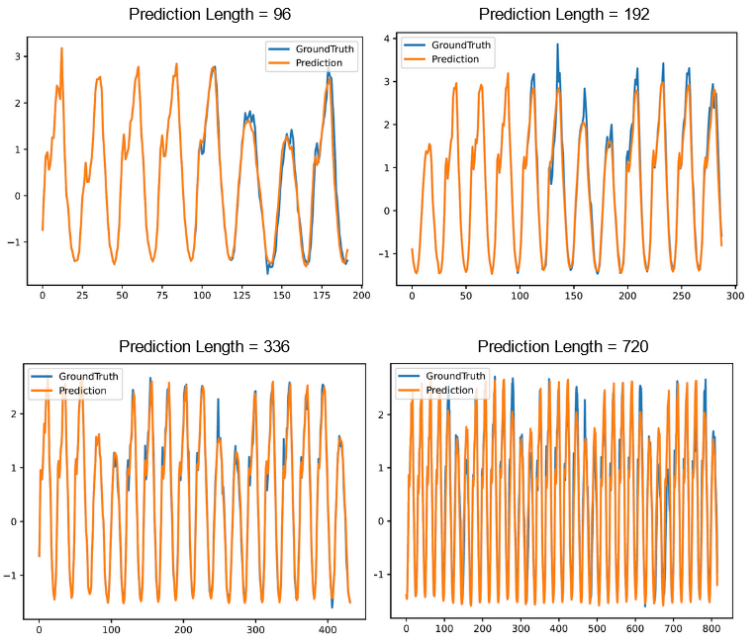


Figure 6.9 – Prediction results of our model applied to the Traffic dataset with different forecasting horizons.

CHAPITRE 7
SHOULD WE RECONSIDER RNNS FOR TIME SERIES FORECASTING ?

This chapter is based on my contribution presented in the paper titled "Should we reconsider RNNs for time series forecasting ?". While Transformer-based models have recently set new standards in time series forecasting, their high computational and memory demands pose some challenges in real-world applications. In this work, I revisit the Gated Recurrent Unit (GRU) as a lightweight and effective alternative. By designing an inverted GRU (iGRU) architecture that combines a feed-forward temporal layer with sequential GRU outputs, the model captures both temporal and cross-channel dependencies efficiently. Experimental results across various benchmark datasets demonstrate that iGRU attains competitive or superior forecasting accuracy with significantly reduced memory consumption compared to Transformer models, highlighting its potential for deployment in resource-constrained environments. While attention-based models are effective, they can be computationally expensive and do not explicitly encode structured dependencies between variables. The intuition behind iGRU is that inter-variable relationships can be captured efficiently by treating variables themselves as a sequence. By applying GRU cells along the channel dimension, the model learns structured dependencies between different indicators. This approach reintroduces recurrent modeling in a targeted manner, significantly reducing computational complexity while preserving predictive performance. Hypotheses 3 and 4 are addressed in this paper by reconsidering RNNs for time series forecasting in a novel way to capture dependencies along the channel dimension. The work presented in this chapter was published in the MDPI AI journal under the title "Should We Reconsider RNNs for Time-Series Forecasting ?". Manuscript writing, conceptualization, model design, implementation, and experiments were performed by Vahid Naghashi, under the supervision of professors Abdoulaye Baniré Diallo and Mounir Boukadoum. A printed version of this paper is attached to the Appendix section.

RÉSUMÉ

In recent years, Transformer-based models have dominated time series forecasting domain, overshadowing recurrent neural networks (RNNs) such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). While Transformers demonstrate superior performance, their high computational cost limits their practical application in resource-constrained settings. In this paper, we reconsider RNNs — specifically the GRU architecture — as an efficient alternative for time series forecasting by leveraging its sequential representation capability to capture cross-channel dependencies effectively. Our model also utilizes a feed-forward layer right after the GRU module to represent temporal dependencies, and aggregates it with a GRU layer to predict future values of a given time series. Our extensive experiments conducted on different real-world datasets show that our inverted GRU (iGRU) model achieves promising results in terms of error metrics and memory efficiency, surpassing state-of-the-art models on various benchmarks.

7.1 Introduction

Time series forecasting is an important task in several application domains, including finance, meteorology, transportation, and energy consumption. Providing accurate forecasts helps industries and businesses save both money and time by enabling optimized and informed decision-making ahead of time, thereby avoiding unnecessary actions. Time series forecasting involves leveraging historical (past) data from various channels or variates to predict future values of the same or related variates. As shown in Figure 7.1, these variables are often inter-correlated, and temporal relationships exist along the time dimension in a time series. Time series analysis and forecasting have garnered significant attention from researchers over the past decades. With the rise of Artificial Intelligence (AI), deep learning-based methods have taken the lead in this field (Chen *et al.*, 2023b). Among the deep learning models developed for time series forecasting, Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), Multi-Layer Perceptrons (MLPs), Transformer models, and Large Language Model (LLM)-based approaches have attained remarkable performance due to their ability to capture complex long-term temporal dependencies (Zhou *et al.*, 2021; Challu *et al.*, 2023). A model usually demonstrates high performance for multivariate time series forecasting when it captures the relations between the prediction variables and the temporal correlations across the historical time steps. There are two mainly used approaches to time series forecasting with deep learning models. The first category of methods are known as Channel-Dependent (CD) methods, which usually project the channel dimension into a hidden space (modeling dimension). However, recent works show that Channel-Independent (CI) models generally achieve better results (Liu *et al.*, 2022; Zeng *et al.*, 2023). Recently, many Transformer-based models have been proposed using channel independence, where multiple channels are predicted independently. In addition, one of the recently invented CI based models, iTransformer (Liu *et al.*, 2023), directly captures the intricate interaction among multiple time series channels by exploiting the high capacity of multi-head self-attention module inside the Transformer, and embedding the temporal dimension to the model dimension, leading to impressive results, specifically in dealing with complex real-world datasets. However, the Transformer-based models face the obvious challenge of quadratic time complexity with respect to the input sequence length, specifically in inference step, which gives rise to an intensive calculation when applied to a large number of variates or longer look-back windows, hindering their deployment in real-world applications.

There have already been attempts to reduce the computational complexity of Transformer for time series forecasting. For instance, (Kitaev *et al.*, 2019) modify the Transformer to focus on a portion of the sequence, while other works utilize linear models to decrease the time complexity (Zeng *et al.*, 2023; Li *et al.*, 2023a).

Although linear models can reduce time complexity, they mainly rely on linear computations and fail to exploit contextual information resulting in sub-optimal predictions, specifically when applied to datasets involving many variates or series. Given the prevalent use of Transformer models in the time series forecasting (TSF) domain, recurrent models such as LSTM and GRU have been neglected and their potential to capture cross-channel dependencies escaped the attention of researchers in the field. Although RNNs are still occasionally utilized for TSF (Lin *et al.*, 2023) in rare cases, however, their capability for sequential modeling is underestimated compared to that of Transformers.

In this paper we reconsider the RNNs, specifically the Gated Recurrent Unit (GRU) as an alternative to the multi-head self-attention module in the Transformer models. We exploit GRUs to extract the interactions among time series channels and utilize them in multiple channel forecasting. Inspired by the iTransformer (Liu *et al.*, 2023) model, we apply the GRU to the channel dimension of the input series to capture inter-series correlations. Considering the importance of capturing cross-channel correlations in time series analysis, we will answer the question of whether RNNs (including GRU or LSTM) should still be considered for time series forecasting ? We answer this question by conducting experiments using various public time series datasets and analyzing the results, specifically through the ablation experiments. Recurrent Neural Networks (RNNs) represent significant advantages in inference time efficiency for time-series forecasting, although their performance sometimes falls behind Transformer-based architectures. The proposed iGRU achieves competitive performance with reduced computational overhead. On datasets like Solar, iGRU outperforms several Transformer models in both accuracy and efficiency, as shown in Section 3. However, for some datasets, iGRU's performance is surpassed by compute-intensive models, reflecting a trade-off between accuracy and computational cost. This work explores these trade-offs, demonstrating iGRU's potential as a lightweight and effective model for time-series forecasting task. Furthermore, in most cases, iGRU outperforms the recently proposed iTransformer (Liu *et al.*, 2023). The prominent reason for utilizing GRUs in our model is that, RNNs provide significantly lower time and memory complexity compared to many Transformer-based models, resulting in improved efficiency for specific applications, as demonstrated in Table 7.1. Additionally, the clear temporal flow in RNNs provides better interpretability in understanding how information propagates through sequences (Hou et Zhou, 2020), specifically when compared to Transformer and MLP-based architectures.

Table 7.1 – Comparison of time and memory complexity for different models, where L represents the input sequence length (context length).

Method	**Type**	**Time Complexity**	**Memory Complexity**
GRU	RNN	$O(L)$	$O(L)$
DLinear	MLP	$O(L)$	$O(L)$
Crossformer	Transformer	$O(L^2)$	$O(L^2)$

Our work includes the following contributions to time series forecasting domain :

— We reconsider RNNs for time series forecasting using a different approach by focusing on the inter-channel dependencies and propose the inverted GRU (iGRU), which exploits GRU blocks to capture interactions between the time series channels and feed-forward layers to represent temporal relations.

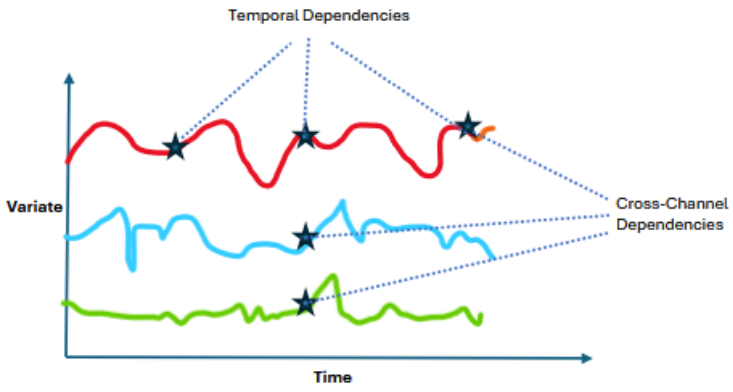


Figure 7.1 – An example of a multivariate time series which consists of multiple channels (variates). There are inter-variate dependencies between the channels and temporal correlations between different time-steps.

— We extensively evaluate iGRU on eleven public datasets and report the results in terms of error metrics and memory efficiency.
— We show that our iGRU model achieves comparable results to the state-of-the-art models or outperforms them.

Time series forecasting methods are generally categorized into statistical models, such as Auto-ARIMA (Contreras *et al.*, 2003), and modern (deep learning) approaches, including Transformer, linear and convolutional neural network models. The Transformer architecture was initially designed to process and generate token sequences, mainly for natural language processing applications, especially Large Language Models (LLMs). However, its excellent potential motivated the TSF research community to deploy and adapt it for time series tasks. For instance, LogTrans (Li *et al.*, 2019) uses convolutional attention in the LogSparse design to capture local information and reduce time complexity. The Informer (Zhou *et al.*, 2021) exploits the ProbSparse self-attention with distillation to emphasize prominent keys. In the Autoformer (Wu *et al.*, 2021), the idea of time series decomposition and auto-correlation calculation is proposed to extract temporal correlations. The FEDformer (Zhou *et al.*, 2022) is designed based on Fourier based architecture and achieves a linear time complexity. In another work, the Pyraformer (Liu *et al.*, 2021b) utilizes pyramidal attention to capture inter-scale and intra-scale relations with a linear complexity. Recently, PatchTST (Nie *et al.*, 2022) is proposed based on a Transformer architecture, which utilizes patched time series with channel independence to capture temporal correlations for each channel, separately. In a different design, the Crossformer (Zhang et Yan, 2023) exploits an encoder-decoder structure with hierarchical attention modules to leverage cross-channel dependencies. However, some Linear models have emerged recently to outperform Transformers in benchmark experiments (Zeng *et al.*, 2023; Das *et al.*, 2023). On the other hand, these linear models fall short in representing non-linear dependencies between the input series and future time steps (Das *et al.*, 2023). Recently, CNN based models achieved promising results in time series analysis tasks. As a prominent example, TimesNet (Wu *et al.*, 2022) utilizes 2-dimensional convolutions to capture inter-period and intra-period relations in a time series with multiple period lengths, obtaining promising results. Over the past few years, Transformers and CNNs have overshadowed Recurrent Neural Networks (RNNs) in the time series forecasting domain. For instance, the iTransformer (Liu *et al.*, 2023) model is proposed based on the vanilla Transformer model and embedding the channels of the input series. This model attained impressive results in many benchmark datasets, pinpointing the importance of modeling cross-channel interactions

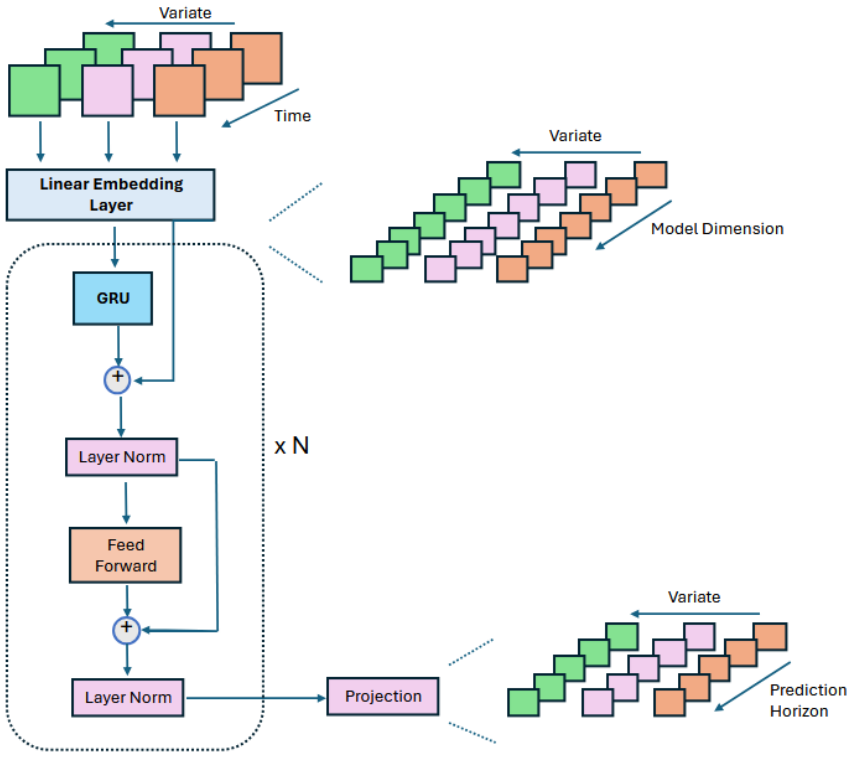


Figure 7.2 – Illustration of iGRU architecture.

in the forecasting tasks. The more recent model, TimeXer (Wang *et al.*, 2024b), incorporates the external information to enhance the forecasting accuracy, which strengthens the canonical Transformer to harmonize endogenous and exogenous information by using patch-wise self-attention and cross-variate attention, simultaneously. In this paper, we reconsider the potential of RNNs for capturing sequential dependencies and introduce the inverted GRU (iGRU), which leverages GRU units to capture dependencies across time series channels while utilizing feed-forward layers to extract temporal features.

7.2 Model Architecture

Our iGRU architecture is illustrated in Figure 7.2 and the corresponding forecasting procedure is shown in Algorithm 1. Given $x_t \in \mathbb{R}^C$, the observation of a time series with C channels at time t, we aim to forecast its future H time-steps $x_{t+1}, ..., x_{t+H}$ using a history or context window w_t of length L (i.e., $w_t = (x_{t-L+1}, ..., x_t)$).

7.2.1 Preliminaries

7.2.1.1 RNN and GRU

Recurrent Neural Networks (RNNs) are a category of neural networks designed for sequential data processing across multiple time steps. The Gated recurring unit (Cho *et al.*, 2014), introduced in 2014, is a specific type of RNN with a gating mechanism to input or forget information along a sequence of timesteps, which

employs update and forget units to process sequential data mapped to a hidden space. The GRU operation is defined by the following equations :

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{7.1}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{7.2}$$

$$\hat{h}_t = \phi(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{7.3}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \tag{7.4}$$

Where, $\odot$ represents element-wise multiplication. x_t and h_t indicate input and hidden state vectors at time t, respectively. z_t, r_t, $\hat{h}_t$ represent the update vector, reset vector and candidate hidden state at time step t respectively. In addition, $W_h, W_z, W_r, U_h, U_z, U_r, b_h, b_z$ and b_r are the relevant weights and biases which are learned during the model training phase. This type of RNN showed effective sequential modeling in various applications (Wang *et al.*, 2021a), (Lawi *et al.*, 2022). In most applications, GRUs and LSTMs are commonly used to capture temporal dependencies. However, their efficiency in modeling cross-channel correlations within time series data has often been overlooked. To leverage RNNs for capturing inter-series dependencies, the input multivariate time series must first be projected into a hidden space using a simple linear embedding layer (Liu *et al.*, 2023). We selected GRU over LSTM due to its comparatively lower parameter count while achieving similar performance, which aligned with our goal of maintaining model efficiency. Vanilla RNNs, on the other hand, are not studied in our approach as they lack memory gates, which limits their performance relative to GRUs.

7.2.1.2 Temporal Embedding of Time Series

Similarly to (Liu *et al.*, 2023), the input multivariate series is embedded into a higher-dimensional space through a linear layer. The input to the temporal embedding has the shape X (Batch, Channel, Time Length), which is an inverted version of the input time series obtained by swapping the temporal and channel dimensions. Then, X is projected into the model space along its temporal dimension by :

$$X_{embedded} = Linear(X) \tag{7.5}$$

Here, $Linear$ refers to a fully connected layer, where the input dimension corresponds to the series length, and the output dimension corresponds to the model dimensionality.

7.2.2 Proposed iGRU Model

As illustrated in Figure 7.2, the input multivariate series is first passed to the linear embedding layer, which is applied series-wise to map each channel into a hidden space. Before embedding, instance normalization (Kim *et al.*, 2021) is utilized to reduce non-stationarity and distribution shifts between the training and test sets. The embedded series is then sent to the GRU module, which acts like the multi-head self-attention of Transformers to capture the intricate interactions among multiple time series channels. Here, the GRU cells represent those dependencies in one direction starting from the first channel to the last one, similar to temporal sequence modeling :

Table 7.2 – Dataset Information : Dim represents the number of variates and Dataset Size denotes the total number of time points in (Train, Validation, Test) split of each dataset, respectively. Prediction Length indicates the future time points to be predicted and four prediction lengths settings are specified in each dataset. Frequency denotes the sampling interval of time points.

Dataset	Dim	Prediction Length	Dataset Size	Frequency	Information
ETTm1, ETTm2	7	{96, 192, 336, 720}	(34465, 11521, 11521)	15min	Electricity
Exchange	8	{96, 192, 336, 720}	(5120, 665, 1422)	Daily	Economy
Weather	21	{96, 192, 336, 720}	(36792, 5271, 10540)	10min	Weather
Electricity	321	{96, 192, 336, 720}	(18317, 2633, 5261)	Hourly	Electricity
Traffic	862	{96, 192, 336, 720}	(12185, 1757, 3509)	Hourly	Transportation
Solar-Energy	137	{96, 192, 336, 720}	(36601, 5161, 10417)	10min	Energy
PEMS03	358	{12, 24, 48, 96}	(15617, 5135, 5135)	5min	Transportation
PEMS04	307	{12, 24, 48, 96}	(10172, 3375, 3375)	5min	Transportation
PEMS07	883	{12, 24, 48, 96}	(16911, 5622, 5622)	5min	Transportation
PEMS08	170	{12, 24, 48, 96}	(10690, 3548, 3548)	5min	Transportation

$$X_{CC} = \overrightarrow{GRU}(X_{embedded}) \tag{7.6}$$

The channel or variate correlated output, X_{CC}, encoded by the GRU layer, is connected with its input to form the output of this layer, facilitating gradient flow and training stability :

$$X_{CCR} = X_{CC} + X_{embedded} \tag{7.7}$$

After adding the GRU output and the skip connection, layer normalization is applied to normalize the activations within each layer to have a mean of zero and a variance of one, thereby stabilizing training through the reduction of varying feature scales. Then, a feed-forward layer (FFN) is applied on the series representations to capture temporal correlations along each channel. The feed-forward network (FFN) consists of two fully-connected layers mapping the series representation to a higher dimensional space (2 times the model dimension) and then to the model dimension, with a Gaussian Error Linear Unit (GELU) activation used between the layers. It is worth noting that FFN implicitly represents temporal dependencies along the model dimension. Finally, the FFN module output is added to its input using a skip connection and a normalization layer is employed afterwards to adjust the obtained series representation. The final prediction is obtained by applying a projection layer to the output of the feed-forward network. The projection layer is a fully-connected layer which projects the model dimension to the forecasting horizon (prediction length), generating the predictions of multiple channels.

Table 7.3 – Selected hyper-parameters for training the iGRU model on different benchmarks.

Dataset	Model Dimension	Feed-Forward Dimension	iGRU Blocks	Learning Rate	Batch Size	Dropout Rate
ETTm1	256	512	2	0.0001	32	0.1
ETTm2	256	512	2	0.0001	32	0.1
Weather	512	512	3	0.0001	32	0.1
Exchange	256	256	2	0.00005	32	0.1
Electricity	512	512	3	0.001	16	0.1
Traffic	512	512	4	0.001	16	0.1
PEMS03	512	512	4	0.001	16	0.1
PEMS04	1024	1024	4	0.001	16	0.1
PEMS07	512	512	3 or 4	0.001	16	0.1
PEMS08	512	512	3 or 4	0.001	16	0.1

7.3 Results

The Proposed iGRU is thoroughly and carefully evaluated on eleven public datasets. The Traffic dataset (Wu *et al.*, 2021) is a collection of road occupancy data from the California Department of Transportation. It was gathered from 862 sensors between 2015 and 2016. PEMS (Liu *et al.*, 2022) is a complex spatio-temporal dataset related to public traffic networks in California consisting of four different datasets (PEMS03, PEMS04, PEMS07, PEMS08). The ETT (Electricity Transformer Temperature) dataset (Zhou *et al.*, 2021) includes data related to load and oil temperature of electricity transformers collected from July 2016 to July 2018. This collection contains different datasets captured hourly or in minutes granularity, including ETTm1 and ETTm2, all consisting of seven variates. The Weather dataset (Wu *et al.*, 2021) consists of 21 meteorological variates recorded every 10 minutes from the Max Planck Bio-geochemistry institute. The Electricity dataset (Wu *et al.*, 2021) contains hourly electricity consumption of 321 costumers. The Solar-Energy dataset (Wu *et al.*, 2021), which was sampled every 10 minutes, collected solar power records in 2006 from 137 PV plants in the US state of Alabama. The Exchange dataset (Wu *et al.*, 2021) is collected based on panel data of daily exchange rates corresponding to eight countries from 1990 to 2016. More information regarding the datasets are reported in Table 7.2.

We compared proposed iGRU model to several state-of-the-art models, including TimeXer (Wang *et al.*, 2024b), iTransformer (Liu *et al.*, 2023), PatchTST (Nie *et al.*, 2022), Crossformer (Zhang et Yan, 2023), TiDE (Das *et al.*, 2023), DLinear (Zeng *et al.*, 2023), FEDformer (Zhou *et al.*, 2022), Autoformer (Wu *et al.*, 2021) and TimesNet (Wu *et al.*, 2022). We implemented our proposed model in Pytorch (Paszke *et al.*, 2019) and executed our experiments using a single A100-40G NVIDIA GPU. All models were trained for 10 epochs with early stopping patience of three steps based on the validation loss change. The Mean Square Error (MSE) loss function is utilized with the Adam (Kingma et Ba, 2014) optimizer to train all models. We set the learning rate to 0.001 for the Traffic, Electricity and PEMS datasets and lower values (0.0005 or 0.0001) are used for the other datasets. This choice is an attempt to mitigate overfitting and skipping of sub-optimal results since these datasets had a limited number of training instances. For each hyper-parameter, we tried a range of possible values, and the value yielding the best result is picked. Reported results represent an average

Table 7.4 – Multivariate time series forecasting results of iGRU and the baseline models on Traffic and PEMS datasets. The input length (lookback window) is set to 96 and prediction length is in {12, 24, 48, 96} for PEMS datasets and in {96, 192, 336, 720} for Traffic dataset.

Models		Ours		TimeXer		iTransformer		PatchTST		DLinear		Crossformer		TimesNet		TiDE		FEDformer		Autoformer	
	Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Traffic	96	**0.393**	**0.268**	0.428	0.271	0.395	0.268	0.462	0.295	0.650	0.396	0.522	0.290	0.593	0.321	0.805	0.493	0.587	0.366	0.613	0.388
	192	**0.417**	*0.277*	0.448	0.282	**0.417**	**0.276**	0.466	0.296	0.598	0.370	0.530	0.293	0.617	0.336	0.756	0.474	0.604	0.373	0.616	0.382
	336	**0.431**	**0.283**	0.473	0.289	0.433	**0.283**	0.482	0.304	0.605	0.373	0.558	0.305	0.629	0.336	0.762	0.477	0.621	0.383	0.622	0.337
	720	**0.463**	**0.301**	0.516	0.307	0.467	0.302	0.514	0.322	0.645	0.394	0.589	0.238	0.640	0.350	0.719	0.449	0.626	0.382	0.660	0.408
	Avg	**0.426**	**0.282**	0.466	0.287	*0.428*	**0.282**	0.481	*0.304*	0.625	0.383	0.550	0.304	0.620	0.336	0.760	0.473	0.610	0.376	0.628	0.379
PEMS03	12	**0.069**	**0.172**	0.072	0.184	*0.071*	*0.174*	0.099	0.216	0.122	0.243	0.090	0.203	0.085	0.192	0.178	0.305	0.126	0.251	0.272	0.385
	24	**0.087**	**0.195**	*0.088*	0.202	0.093	*0.201*	0.142	0.259	0.201	0.317	0.121	0.240	*0.118*	0.223	0.257	0.371	0.149	0.275	0.334	0.440
	48	**0.119**	**0.230**	0.127	0.242	*0.125*	*0.236*	0.211	0.319	0.333	0.425	0.202	0.317	0.155	*0.260*	0.379	0.463	0.227	0.348	1.032	0.782
	96	**0.151**	**0.264**	0.177	0.284	*0.164*	*0.275*	0.269	0.370	0.457	0.515	0.262	0.367	*0.228*	*0.317*	0.490	0.539	0.348	0.434	1.031	0.796
	Avg	**0.107**	**0.215**	0.116	0.228	*0.113*	*0.221*	0.180	0.291	0.278	0.375	0.169	0.281	0.147	0.248	0.326	0.419	0.213	0.327	0.667	0.601
PEMS04	12	**0.078**	*0.185*	0.082	0.197	**0.078**	**0.183**	0.105	0.224	0.148	0.272	0.098	0.218	0.087	0.195	0.219	0.340	0.138	0.262	0.424	0.491
	24	**0.091**	**0.204**	0.094	0.212	*0.095*	*0.205*	0.153	0.275	0.224	0.340	0.131	0.256	0.103	0.215	0.292	0.398	0.177	0.293	0.459	0.509
	48	**0.114**	**0.230**	0.119	0.237	*0.120*	*0.233*	0.229	0.339	0.355	0.437	0.205	0.326	0.136	0.250	0.409	0.478	0.270	0.368	0.646	0.610
	96	**0.141**	**0.254**	0.162	0.275	*0.150*	*0.262*	0.291	0.389	*0.452*	*0.504*	0.402	0.457	0.190	0.303	0.492	0.532	0.341	0.427	0.912	0.748
	Avg	**0.106**	**0.218**	0.114	0.230	*0.111*	*0.221*	0.195	0.307	0.295	0.388	0.209	0.314	0.129	0.241	0.353	0.437	0.231	0.337	0.610	0.590
PEMS07	12	*0.065*	**0.163**	**0.063**	0.171	0.067	*0.165*	0.095	0.207	0.115	0.242	0.094	0.200	0.082	0.181	0.173	0.304	0.109	0.225	0.199	0.336
	24	*0.084*	*0.188*	**0.079**	**0.187**	0.088	0.190	0.150	0.262	0.210	0.329	0.139	0.247	0.101	0.204	0.271	0.383	0.125	0.244	0.323	0.420
	48	*0.103*	*0.210*	**0.100**	**0.203**	0.110	0.215	0.253	0.340	0.398	0.458	0.311	0.369	0.134	0.238	0.446	0.495	0.165	0.288	0.390	0.470
	96	**0.128**	*0.235*	*0.131*	**0.233**	0.139	0.245	0.346	0.404	0.594	0.553	0.396	0.442	0.181	0.279	0.628	0.577	0.262	0.376	0.554	0.578
	Avg	*0.095*	**0.199**	**0.093**	**0.199**	0.101	0.204	0.211	0.303	0.329	0.395	0.235	0.315	0.193	0.271	0.380	0.440	0.165	0.283	0.367	0.451
PEMS08	12	**0.077**	**0.179**	0.091	0.206	*0.079*	*0.182*	0.168	0.232	0.154	0.276	0.165	0.214	0.112	0.212	0.227	0.343	0.173	0.273	0.436	0.485
	24	**0.109**	**0.212**	0.133	0.253	*0.115*	*0.219*	0.224	0.281	0.248	0.353	0.215	0.260	0.141	0.238	0.318	0.409	0.210	0.301	0.467	0.502
	48	**0.177**	**0.232**	0.209	0.249	*0.186*	*0.235*	0.321	0.354	0.440	0.470	0.315	0.355	0.198	0.283	0.497	0.510	0.320	0.394	0.966	0.733
	96	**0.213**	**0.262**	0.492	0.467	*0.221*	*0.267*	0.408	0.417	0.674	0.565	0.377	0.397	0.320	0.351	0.721	0.592	0.442	0.465	1.385	0.915
	Avg	**0.144**	**0.221**	0.231	0.294	*0.150*	*0.226*	0.280	0.321	0.379	0.416	0.268	0.307	0.193	0.271	0.441	0.464	0.286	0.358	0.814	0.659

of five runs. In our experiments, the batch size is uniformly selected as 16 or 32, and the number of iGRU blocks are set from $\{1, 2, 3, 4\}$. Additionally, model dimension is chosen from $\{128, 256, 512\}$ according to each dataset. The selected hyper-parameters for each dataset are shown in Table 7.3.

7.4 Discussion

The results of time series forecasting in terms of MSE and MAE (Mean Absolute Error) are reported in Tables 7.4 and 7.5. Our iGRU model outperforms most baseline models, notably iTransformer, by capturing inter-series relations with GRU modules. Despite a relatively low number of variates in the ETT datasets (7 variates), iGRU shows better performance compared to iTransformer and PatchTST models. For instance, on the ETTm1 dataset our iGRU model outperforms iTransformer by more than 4% on average in terms of MSE metric, highlighting its efficiency in capturing relations between variates and temporal time steps. Our model efficiency is also verified by its performance on datasets with numerous periodic variates, including

Algorithm 1 The forecasting procedure of iGRU

Input : Batch$(X) = [x_1, x_2, \ldots, x_L] : (B, L, C)$

Output : Batch$(Y) = [y_1, y_2, \ldots, y_H] : (B, H, C)$

$X_T : (B, C, L) \leftarrow$ Transpose(Batch(X))
$X_{embedded} : (B, C, D) \leftarrow$ Embedding(X_T)
for each layer l in iGRU layers **do**
    $X_{CC} \leftarrow \overrightarrow{\text{GRU}}(X_{embedded})$
    $X_{CCR} \leftarrow X_{CC} + X_{embedded}$
    $X_{CCR} \leftarrow$ LayerNorm(X_{CCR})
    $X_{FF} \leftarrow$ Feed-Forward(X_{CCR})
    $X_{FF} \leftarrow X_{FF} + X_{CCR}$
    $X_F \leftarrow$ LayerNorm(X_{FF})
    $X_{embedded} \leftarrow X_F$
end for
$X_{out} : (B, C, H) \leftarrow$ Projection(X_F)
Batch$(Y) : (B, H, C) \leftarrow$ Transpose(X_{out})

Traffic, Electricity, and PEMS datasets. As observed in Table 7.4, the iGRU model exhibits noticeably higher MSE and MAE values on the Traffic dataset compared to the PEMSn datasets. This difference can be attributed to the inherent complexity and variability of the Traffic dataset, which includes diverse traffic patterns influenced by external factors such as weather, events, or road conditions, making it less predictable than the PEMSn datasets. In addition, the Traffic dataset's hourly sampling frequency records broader trends, amplifying variability and reducing short-term pattern consistency, whereas the PEMSn datasets' 5-minute sampling pattern provides finer patterns, facilitating more predictable temporal dependencies. This trend of elevated errors is consistent across other models evaluated on the traffic dataset, suggesting that the dataset's characteristics pose a general challenge. In particular, the results associated with Traffic, PEMS03, PEMS04, PEMS07 and PEMS08 datasets underscore the capability of iGRU in handling inter-series dependencies more efficiently, compared to other baseline models. According to Table 7.4, iGRU reduces the MSE error (averaged over four prediction lengths) by nearly 6% compared to the second best baseline (iTransformer) on the PEMS07 dataset, which comprises 883 variates and represents a spatio-temporal type of time series. iGRU also achieves prediction accuracy comparable to TimeXer while consuming less GPU memory and training time, especially when prediction length is set to 96. On the Traffic dataset, iGRU reduces the MSE error by more than 8.5% on average, relative to TimeXer. When averaged across four prediction horizons, MSE error also improved by more than 21% on the PEMS08 dataset compared to TimeXer. This highlights the capability of iGRU model to capture complex cross-variate dependencies. To demonstrate the robustness of our iGRU model, we trained it five times with various random seeds and reported the mean and standard deviation of the results in Table 7.7. The low standard deviations of MSE and MAE errors across the test sets associated with different datasets confirm the stability and robustness of the iGRU model.

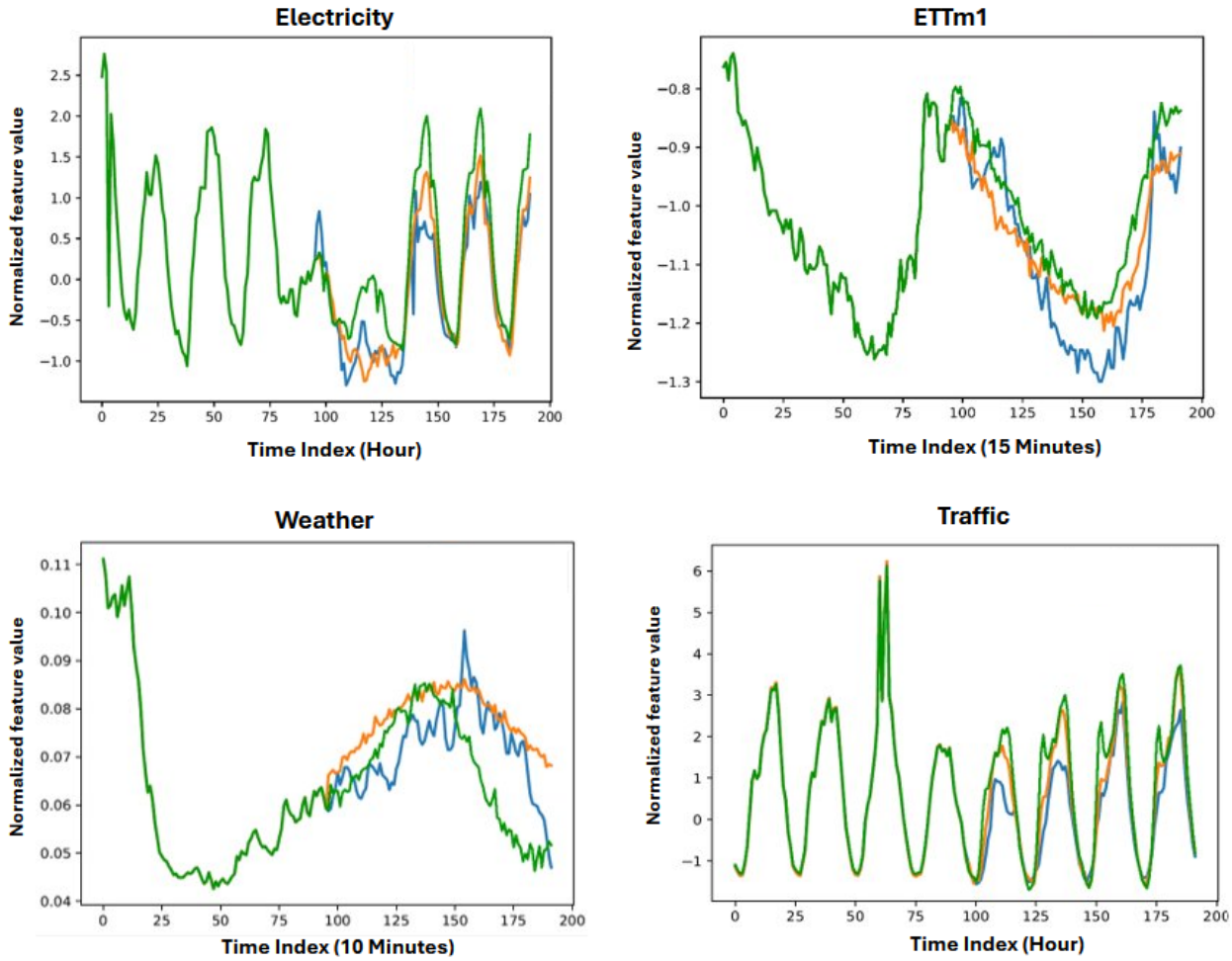


Figure 7.3 – Visual comparison of iGRU and iTransformer on different datasets. Orange line indicates predictions for iGRU and green line corresponds to predictions for iTransformer, with blue line indicating the ground truth. The look-back window and forecast window lengths are set to 96 for all datasets.

To illustrate the performance of iGRU on different datasets intuitively, we present a visual comparison of its predictions against the ground truth target series. These visualizations provide a clear and interpretable assessment of the model forecasting accuracy. In the provided plots (Figure 7.3), the orange line indicates the predictions related to a model and the blue line demonstrates the actual selected sequence. Figure 7.3 indicates that predictions corresponding to the iGRU are well-aligned with the ground truth series on different datasets compared to the predictions generated by the iTransformer.

7.4.1 Model Efficiency

To assess our model's computational efficiency, its memory consumption and training time are compared against those of the other baselines on the Traffic and PEMS07 datasets. Here, by efficiency we mean training time and GPU memory consumption. Independent runs are conducted using a single A100-40G GPU with the batch size fixed to 16. Our model's efficiency is illustrated in Figure 7.4, where bubble charts show

Table 7.5 – Multivariate time series forecasting results of iGRU and the baseline models on Weather, Electricity, Exchange, Solar-energy, ETTm1 and ETTm2 datasets. The input length (look-back window) is set to 96 and prediction length is in {96, 192, 336, 720}. The best results and second best results are indicated in bold and italics, respectively.

Models		**Ours**		**TimeXer**		**iTransformer**		**PatchTST**		**DLinear**		**Crossformer**		**TimesNet**		**TiDE**		**FEDformer**		**Autoformer**	
	Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	*0.167*	0.210	**0.157**	**0.205**	0.174	*0.214*	0.177	0.218	0.196	0.255	*0.158*	0.230	0.172	0.220	0.202	0.261	0.217	0.296	0.266	0.336
	192	0.215	*0.254*	**0.204**	**0.247**	0.221	*0.254*	0.225	0.259	0.237	0.296	*0.206*	0.277	0.219	0.261	0.242	0.298	0.276	0.336	0.307	0.367
	336	0.273	*0.297*	**0.261**	**0.290**	0.278	*0.296*	0.278	0.297	0.283	0.335	*0.272*	0.335	0.280	0.306	0.287	0.335	0.339	0.380	0.359	0.395
	720	0.354	0.349	**0.340**	**0.341**	0.358	*0.347*	0.354	0.348	*0.345*	0.381	0.398	0.418	0.365	0.359	0.351	0.386	0.403	0.428	0.419	0.428
	Avg	*0.253*	*0.278*	**0.241**	**0.271**	0.258	*0.278*	0.259	0.281	0.265	0.317	0.259	0.315	0.259	0.287	0.271	0.320	0.309	0.360	0.338	0.382
Electricity	96	*0.142*	**0.238**	**0.140**	0.242	0.148	*0.240*	0.181	0.270	0.197	0.282	0.219	0.314	0.168	0.272	0.237	0.329	0.193	0.308	0.201	0.317
	192	*0.160*	*0.255*	**0.157**	0.256	0.162	**0.253**	0.188	0.274	0.196	0.285	0.231	0.322	*0.184*	0.289	0.236	0.330	0.201	0.315	0.222	0.334
	336	**0.176**	*0.272*	**0.176**	0.275	*0.178*	**0.269**	0.204	0.293	0.209	0.301	0.246	0.337	0.198	*0.300*	0.249	0.344	0.214	0.329	0.231	0.338
	720	**0.210**	**0.301**	*0.211*	*0.306*	0.225	0.317	0.246	0.324	0.245	0.333	0.280	0.363	*0.220*	*0.320*	0.284	0.373	0.246	0.355	0.254	0.361
	Avg	*0.172*	**0.267**	**0.171**	0.270	0.178	*0.270*	0.205	0.290	0.212	0.300	0.244	0.334	0.192	0.295	0.251	0.344	0.214	0.327	0.227	0.338
Solar	96	*0.193*	*0.245*	**0.187**	0.250	0.203	**0.237**	0.234	0.286	0.290	0.378	0.310	0.331	0.250	0.292	0.312	0.399	0.242	0.342	0.884	0.711
	192	**0.199**	**0.253**	*0.202*	0.271	0.233	*0.261*	0.267	0.310	0.320	0.398	0.734	0.725	0.296	0.318	0.339	0.416	0.285	0.380	0.834	0.692
	336	**0.203**	**0.257**	*0.215*	0.284	0.248	*0.273*	0.290	0.315	0.353	0.415	0.750	0.735	0.319	0.330	0.368	0.430	0.282	0.376	0.941	0.723
	720	**0.207**	**0.259**	*0.220*	0.293	0.249	*0.275*	0.289	0.317	*0.356*	*0.413*	0.769	0.765	0.338	0.337	0.370	0.425	0.357	0.427	0.882	0.717
	Avg	**0.201**	**0.254**	*0.229*	0.274	0.233	*0.262*	0.270	0.307	0.330	0.401	0.641	0.639	0.301	0.319	0.347	0.417	0.291	0.381	0.885	0.711
Exchange	96	**0.086**	0.207	**0.086**	*0.206*	**0.086**	*0.206*	*0.088*	**0.205**	*0.088*	0.218	0.256	0.367	0.107	0.234	0.094	0.218	0.148	0.278	0.197	0.323
	192	*0.181*	*0.304*	0.188	0.308	0.177	**0.299**	**0.176**	**0.299**	**0.176**	0.315	0.470	0.509	0.226	0.344	0.184	0.307	0.271	0.315	0.300	0.369
	336	*0.331*	*0.417*	0.342	0.421	0.331	*0.417*	**0.301**	**0.397**	0.313	0.427	1.268	0.883	0.367	0.448	0.349	0.431	0.460	0.427	0.509	0.524
	720	0.857	0.702	0.870	0.702	*0.847*	**0.691**	0.901	0.714	**0.839**	*0.695*	1.767	1.068	0.964	0.746	0.852	0.698	1.195	0.695	1.447	0.941
	Avg	0.364	0.408	0.372	0.409	*0.360*	**0.403**	0.367	*0.404*	**0.354**	0.414	0.940	0.707	0.416	0.443	0.370	0.413	0.519	0.429	0.613	0.539
ETTm1	96	*0.321*	*0.358*	**0.318**	**0.356**	0.334	0.368	0.329	0.367	0.345	0.372	0.404	0.426	0.338	0.375	0.364	0.387	0.379	0.419	0.505	0.475
	192	*0.364*	**0.382**	**0.362**	*0.383*	0.377	0.391	0.367	0.385	0.380	0.389	0.450	0.451	0.374	0.387	0.398	0.404	0.426	0.441	0.553	0.496
	336	*0.399*	**0.406**	**0.395**	*0.407*	0.426	0.420	*0.399*	0.410	0.413	0.413	0.532	0.515	0.410	0.411	0.428	0.425	0.445	0.459	0.621	0.537
	720	0.470	0.445	**0.452**	*0.441*	0.491	0.459	*0.454*	**0.439**	0.474	0.453	0.666	0.589	0.478	0.450	487	0.461	0.543	0.490	0.671	0.561
	Avg	0.389	*0.398*	**0.382**	**0.397**	0.407	0.410	*0.387*	0.400	0.403	0.407	0.513	0.496	0.400	0.406	0.419	0.419	0.448	0.452	0.588	0.517
ETTm2	96	0.177	0.260	**0.171**	**0.256**	0.180	0.264	*0.175*	*0.259*	0.193	0.292	0.287	0.366	0.187	0.267	0.207	0.305	0.203	0.287	0.255	0.339
	192	0.242	0.304	**0.237**	**0.299**	0.250	0.309	*0.241*	*0.302*	0.284	0.362	0.414	0.492	0.249	0.304	0.290	0.364	0.269	0.328	0.281	0.340
	336	0.306	*0.343*	**0.296**	**0.338**	0.311	0.348	*0.305*	*0.343*	0.369	0.427	0.597	0.542	0.321	0.351	0.377	0.422	0.325	0.366	0.339	0.372
	720	0.408	*0.406*	**0.392**	**0.394**	0.412	0.407	*0.402*	0.412	0.554	0.522	1.730	1.042	0.408	0.403	0.558	0.524	0.421	0.415	0.433	0.432
	Avg	0.283	0.328	**0.274**	**0.322**	0.288	0.332	*0.281*	*0.326*	0.350	0.401	0.757	0.610	0.290	0.333	0.358	0.404	0.305	0.349	0.327	0.371

a visual comparison of efficiency metrics. The vertical axis indicates the prediction MSE, and the horizontal axis depicts the duration of one training iteration (milli-seconds/iteration). The bubble size indicates the related memory footprint in Gigabytes (total memory consumption in one epoch). As Figure 7.4 illustrates, the iGRU model attains the most accurate results, while requiring less or equal training time and memory usage compared to other baselines, except the linear models. DLinear consumes minimal memory and time resources than the other models, while delivering the least accurate forecasts. To further support the claim regarding the computational efficiency of iGRU compared to Transformer-based models, we conducted additional experiments on the Electricity dataset with varying input lengths (96, 336, and 720), while keeping the prediction length fixed at 96. For each configuration, we measured the training time (ms/iteration),

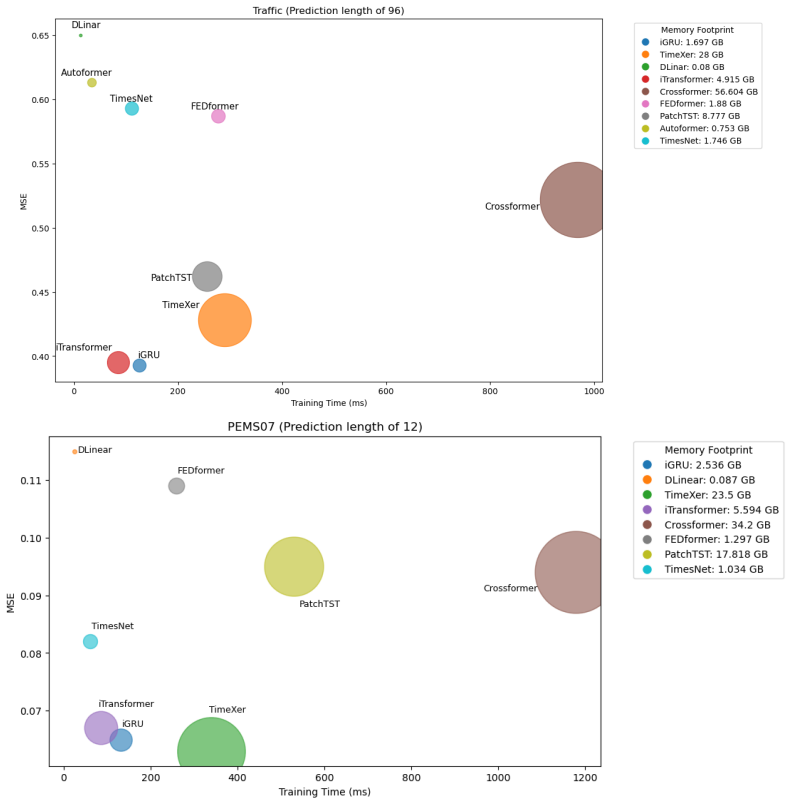
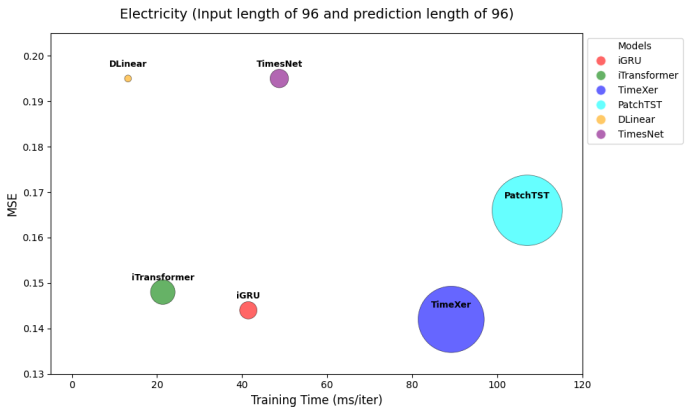


Figure 7.4 – Comparison of iGRU with other baselines in terms of MSE, training time and GPU memory usage on Traffic and PEMS07 datasets. Look-back window is set to 96 and prediction length is set to 96 and 12 for Traffic and PEMS07 datasets, respectively.

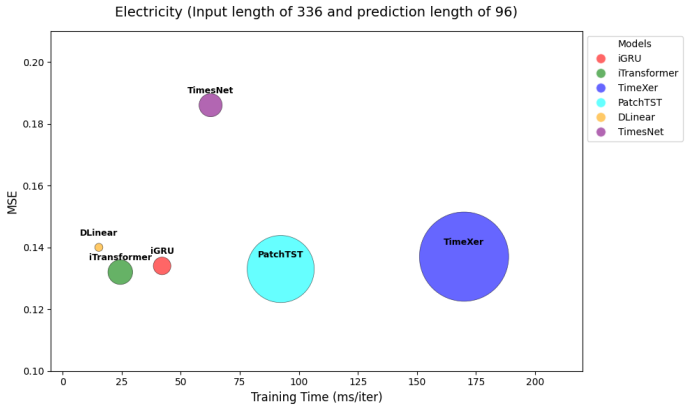
peak GPU memory usage, and related model performance (MSE). Figure 7.5 illustrates how model cost vary across different input lengths. Notably, iGRU consistently maintains low training time and memory footprint, while delivering competitive or better accuracy compared to Transformer-based alternatives. These results confirm that iGRU offers a stable trade-off between efficiency and performance, specifically when the optimal model is sensitive to the length of input window.

7.4.2 Ablation Study

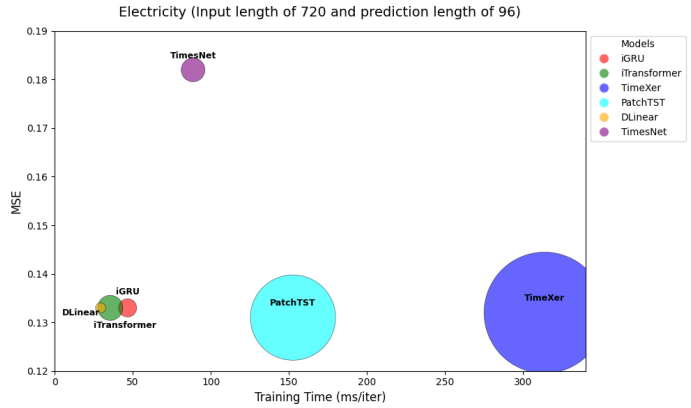
We demonstrate the importance of the RNN (GRU) and feed-forward layers in our model through conducting experiments with and without the GRU, feed-forward layer and skip connections. The results are reported in Table 7.6, showcasing the impact of these components on the iGRU performance. The impact of the above components is mostly evident on the Traffic dataset which is a complex dataset with many time series variates. The skip connections added to the outputs of the GRU and feed-forward layers contribute to the gradient flow and training efficiency, which helps to attain optimal results. The significant contribution of GRU blocks in capturing cross-variate correlations becomes evident when comparing the results of the model without GRU blocks (W/O GRU) to those of the iGRU model, which incorporates these blocks. This comparison underscores the potential of GRUs to enhance time series forecasting performance and



(a) Input length : 96



(b) Input length : 336



(c) Input length : 720

Figure 7.5 – Comparison of training time (ms/iter), model performance (MSE), and memory usage (bubble size) across different input lengths on the Electricity dataset. iGRU consistently demonstrates low training cost and memory usage while maintaining competitive or better performance compared to Transformer-based models.

Table 7.6 – Ablation of iGRU model without GRU, skip or residual connections and feed-forward layer.

Design		W/O F.F		W/O F.F + W/O Skip Connection		W/O Skip Connection		W/O GRU		iGRU	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	0.324	0.361	0.325	0.364	0.327	0.364	0.324	0.362	**0.321**	**0.358**
	192	0.366	0.383	0.368	0.387	0.373	0.390	0.367	0.384	**0.364**	**0.382**
	336	**0.399**	**0.405**	0.403	0.410	0.415	0.418	0.400	0.406	**0.399**	0.406
	720	**0.467**	**0.443**	0.475	0.450	0.482	0.455	**0.467**	0.444	0.470	0.445
Traffic	96	0.407	0.281	0.414	0.287	0.502	0.366	0.437	0.282	**0.393**	**0.268**
	192	0.428	0.289	0.437	0.296	0.535	0.372	0.450	0.287	**0.417**	**0.277**
	336	0.445	0.296	0.452	0.303	0.537	0.371	0.464	0.294	**0.431**	**0.283**
	720	0.476	0.313	0.487	0.323	0.572	0.389	0.495	0.312	**0.463**	**0.301**
Weather	96	0.170	0.214	**0.166**	**0.211**	0.169	0.213	0.194	0.232	0.168	**0.211**
	192	0.217	0.256	**0.214**	0.255	0.217	0.257	0.239	0.269	0.215	**0.254**
	336	0.274	0.297	**0.271**	**0.296**	0.277	0.300	0.291	0.307	0.274	0.297
	720	**0.353**	0.349	**0.353**	0.349	0.357	0.352	0.364	0.354	0.354	**0.348**

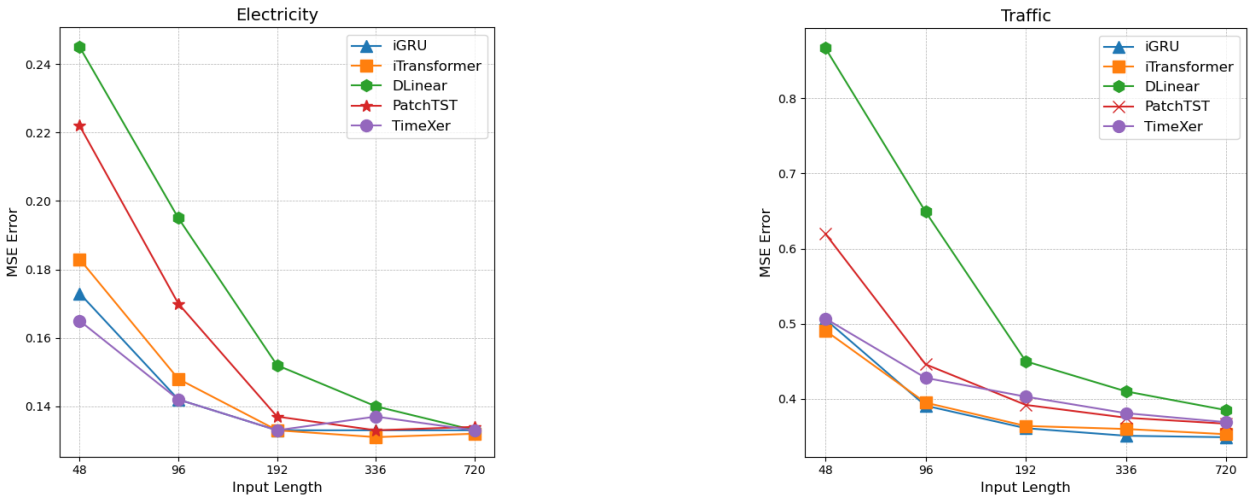
prompts a raised discussion on whether RNNs, particularly GRUs, should be reconsidered as a powerful tool in time series analysis, specifically for modeling cross-channel dependencies.

7.4.3 Increasing look-back length

Some of the previous Transformer based works (Nie *et al.*, 2022; Zeng *et al.*, 2023) have shown that increasing the length of look-back window does not necessarily improve the forecasting results, which can be caused by distracted attention on the increasing (prolonged) input. As the structure of our iGRU model is different from the Transformer based models, we evaluate the performance of iGRU and its main competitor models, e.g. TimeXer, iTransformer, DLinear and PatchTST in Figure 7.6 using various input lengths. The results pinpoint the performance promotion of iGRU for longer input windows and its capability to leverage the information over the extended temporal context.

7.5 Conclusion

In this work, recurrent neural network architecture has been brought back to the time series forecasting field by using it in a different manner and integrating it with a feed-forward layer. Experimentally, proposed iGRU attains competitive results to those of the other state-of-the-art models, while consuming less training time and memory. As future work, we will investigate large-scale foundation models using the iGRU and perform



(a) (b)

Figure 7.6 – Forecasting with look-back length in {48, 96, 192, 336, 720} and prediction length of 96 on the Electricity and Traffic datasets. Proposed iGRU model exploits enlarged and shortened input lengths and delivers accurate results.

Table 7.7 – Robustness of iGRU model. Five independent runs are conducted with different random seeds.

Dataset	**Electricity**		**Traffic**		**Weather**	
Horizon	MSE	MAE	MSE	MAE	MSE	MAE
96	0.142 -/+ 0.000	0.238 -/+ 0.000	0.393 -/+ 0.001	0.267 -/+ 0.001	0.167 -/+ 0.002	0.210 -/+ 0.001
192	0.160 -/+ 0.000	0.254 -/+ 0.000	0.416 -/+ 0.000	0.277 -/+ 0.001	0.215 -/+ 0.000	0.254 -/+ 0.001
336	0.176 -/+ 0.000	0.272 -/+ 0.000	0.431 -/+ 0.000	0.283 -/+ 0.000	0.273 -/+ 0.000	0.297 -/+ 0.000
720	0.210 -/+ 0.003	0.301 -/+ 0.003	0.463 -/+ 0.000	0.301 -/+ 0.000	0.354 -/+ 0.001	0.349 -/+ 0.001
Dataset	**ETTm1**		**ETTm2**		**Exchange**	
Horizon	MSE	MAE	MSE	MAE	MSE	MAE
96	0.320 -/+ 0.001	0.358 -/+ 0.001	0.178 -/+ 0.000	0.260 -/+ 0.000	0.086 -/+ 0.000	0.207 -/+ 0.000
192	0.364 -/+ 0.000	0.382 -/+ 0.000	0.244 -/+ 0.001	0.304 -/+ 0.001	0.181 -/+ 0.000	0.304 -/+ 0.000
336	0.398 -/+ 0.000	0.405 -/+ 0.000	0.304 -/+ 0.001	0.343 -/+ 0.000	0.331 -/+ 0.000	0.417 -/+ 0.000
720	0.469 -/+ 0.001	0.445 -/+ 0.000	0.408 -/+ 0.001	0.403 -/+ 0.001	0.857 -/+ 0.000	0.702 -/+ 0.000

a more in depth exploration of time series analysis.

CHAPITRE 8
A BI-DIRECTIONAL CROSS-CHANNEL RNN MODEL FOR TIME-SERIES FORECASTING OF DAIRY PRODUCTION

This chapter is based on an accepted scientific article, which focuses on forecasting cumulative milk production in dairy cattle through multivariate time series modeling. The prediction task is framed as a multivariate time series forecasting problem that incorporates a variety of dairy-related features, such as health indicators, milk quality, quantity and seasonal patterns. To effectively represent the complex interdependencies between these variables, I designed a recurrent neural network architecture based on Gated Recurrent Units (GRUs) that operates bidirectionally along the channel dimension. The so-called channel-aware design allows the model to efficiently capture cross-feature relationships, while feed-forward layers integrated into the architecture implicitly model temporal dependencies along the model space. Interactions between variables are often directional and reciprocal, meaning that the influence of one variable on another is not symmetric. The intuition behind the bidirectional cross-channel architecture is that modeling dependencies in a single direction may overlook important relationships. By processing variables in both forward and reverse orders, the model captures complementary interaction patterns. This bidirectional design completes the cross-channel modeling introduced previously and leads to richer representations of multivariate dependencies. Experimental evaluations demonstrate that the model achieves competitive or even superior forecasting performance, particularly in predicting cumulative milk income across different lactation periods, while maintaining a lower computational footprint than recent Transformer- and Mamba-based approaches. The work presented in this chapter has been accepted in the Nature Scientific Reports journal, under the title “A bi-directional cross-channel RNN model for time series forecasting of dairy production.” This chapter addresses Hypotheses 1, 2, and 4 of this thesis. A printed version of this paper is attached to Appendix E.

RÉSUMÉ

Predicting milk production in dairy cattle is essential for precision livestock management, a goal that can be achieved by analyzing historical cow data, including health status, milk quality, and seasonal effects. This challenge is framed as a multivariate time-series forecasting problem, requiring the effective capture of temporal dependencies and interrelationships among dairy-related variables (features). Existing time-series models often struggle to adequately represent these intricate dynamics. To address this, we propose a recurrent neural network (RNN) architecture that leverages Gated Recurrent Units (GRUs) applied bidirectionally along the channel dimension to model these interactions efficiently. The proposed model is further enhanced with feed-forward layers to implicitly capture temporal dependencies. Our approach delivers superior or competitive performance, particularly in terms of various error metrics, compared to state-of-the-art methods when predicting cumulative milk income across different lactation periods. Additionally, it exhibits relatively lower time complexity than recently proposed Transformer and convolution-based models.

8.1 Introduction

Over the past few decades, global food consumption has increased significantly, creating a demand for farmers to enhance their production and farming tools (Bruinsma, 2017). Dairy farming, as a key agricultural sector, stands out with its substantial contribution to milk production. Recognizing the vital role of the dairy industry in both human life and agriculture, it is of great importance to equip farmers with the latest technologies to achieve efficient milk production. (Frasco *et al.*, 2020). Considering the great success of artificial intelligence in different fields, leveraging deep learning methods to offer reliable predictions regarding calving, diseases, and productivity enables farmers to make informed decisions about their livestock management and implement strategic measures to optimize their herd conditions (Borchers *et al.*, 2017). The problem of estimating dairy milk yield can be stated as a multivariate time series forecasting task, based on the multiple and heterogeneous dairy features recorded periodically on test days during the lactation months. Dairy features are categorized into health, management, milk quality, milk quantity, farm environmental, and seasonal factors. In this paper, we propose a model to predict milk income throughout the upcoming lactation, taking into account the information from previous lactation period(s). This study builds upon previous works on dairy profitability prediction (Naghashi *et al.*, 2023; Frasco *et al.*, 2020), and aims to extend them in several important ways. Prior models primarily focused on a limited lactation period span, often ignoring the heterogeneous nature of dairy production series where milk yield, animal health, management factors, and seasonal effects interact in complex patterns. Despite their success, recently proposed architectures exhibit conceptual limitations in modeling these intricate cross-channel dynamics. Transformer-based models such as PatchTST (Nie *et al.*, 2022) capture temporal dependencies within each channel by relying on time series patchifying and attention that may not explicitly model the sequential information across channels. MLP-Mixer models (Wang *et al.*, 2024b), while computationally efficient, decompose and mix features across different scales, an operation that might risk obscuring the ordered, non-commutative nature of biological processes. Despite the great performance delivered by State Space Models like S-Mamba (Wang *et al.*, 2025) in sequential modeling, the potential of simpler architectures like RNNs, when applied in a different manner, has been largely underexplored. This lack of simple models of efficient sequence processing represents a significant gap in current research. To address it, our approach repurposes a Recurrent Neural Network to model dependencies sequentially along the channel dimension. Instead of processing time steps, our model treats the time series variables themselves as an ordered sequence and uses a Gated Recurrent Unit to capture their relationships. This conceptual change allows our model to directly learn the intricate interplay between various dairy like health, management, and produc-

tion. Our primary contribution is therefore not the invention of a time series model, but the demonstration that a novel application of a classic architecture can outperform more complex state-of-the-art methods in domains where cross-channel dynamics matter. The forecast generated by our model can offer valuable assistance for dairy farmers. Our designed model is based on the well-known Recurrent Neural Network (RNN), specifically the Gated Recurrent Unit (GRU) (Cho *et al.*, 2014), with modifications to capture cross-channel dependencies among sequences of dairy variables recorded on specific test days across lactation periods. This work makes the following contributions to the state of the art :

— A bidirectional cross-channel model is proposed to predict dairy production from multivariate time series associated with multiple dairy variables recorded along the lactation periods.
— Our designed model captures both temporal and inter-series (cross-channel) correlations, efficiently.
— The proposed model is used to predict milk production factors for dairy cattle in different lactation periods and the results are compared with the baseline models.

8.2 Related Work

In recent years, the dairy industry has witnessed a proliferation of deep learning-based approaches. The Long-Short Term Memory (LSTM) neural network has been utilized for prediction of milk profitability (Frasco *et al.*, 2020), which relies on an integrated multiple source test-date information to predict profits in future months. A deep learning framework is proposed in (Liseune *et al.*, 2021) to encode dairy features, predict the latent representation of milk yield sequences, and generate the lactation curve by using Convolutional Neural Networks (CNN), encoder, and decoder modules. The prediction of first test day milk yield is accomplished through the application of classical machine learning models, including random forest, in (Salamone *et al.*, 2022). The authors in (Naghashi *et al.*, 2023) proposed a hybrid model based on LSTM and attention modules to predict milk income along the second lactation of dairy cattle. Models for time series forecasting can be generally categorized into conventional statistical methods and deep learning models. Statistical models, such as ARIMA (Auto-regressive Integrated Moving Average) have shown good performance in various forecasting applications (Contreras *et al.*, 2003). N-BEATS (Oreshkin *et al.*,) is a well-known deep learning model that is particularly suited for univariate time series forecasting. It uses backward and forward residual connections along with a deep stack of fully connected layers, making it fast to train while also being interpretable. The Transformer architecture has been successful in processing sequences and predicting time-series values. Over the past few years, several time series forecasting models have been proposed that rely on Transformer building blocks. For example, a deep Transformer model with encoder and decoder was introduced in (Wu *et al.*, 2020a) to predict the prevalence of the influenza disease. Informer, a Transformer variant with probSparse self-attention in (Zhou *et al.*, 2021) exploits a specialized attention mechanism and a distillation process to emphasize dominant scores, reducing computational overhead and memory usage. In addition, other Transformer based models have been developed, including Autoformer (Wu *et al.*, 2021), FEDformer (Zhou *et al.*, 2022), Crossformer (Zhang et Yan, 2022), PatchTST (Nie *et al.*, 2022), and iTransformer (Liu *et al.*, 2023). The well-known PatchTST model (Nie *et al.*, 2022) pioneered the use of patch embedding and applied self-attention to time series patches, marking a significant advancement in time series forecasting tasks. On the other hand, MLP models utilize linear layers to forecast future values of time series. DeepAR (Salinas *et al.*, 2020) employs MLP in an auto-regressive fashion to make probabilistic predictions. DLinear (Zeng *et al.*, 2023) decomposes time series into trend and seasonality components, uses MLPs to generate the corresponding future trend and seasonality components and then combines them to deliver the final predictions. In another related work, Rlinear (Li *et al.*, 2023a) introduces a novel method by applying a linear mapping prior to the linear layers, effectively transforming

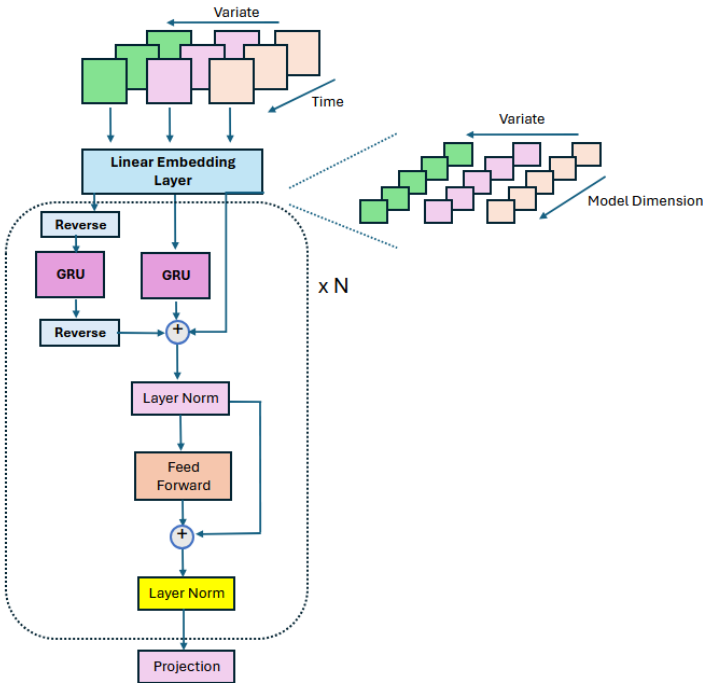


Figure 8.1 – An illustration of proposed bidirectional cross-channel GRU (Bi-iGRU) model.

the input space to improve the model's performance in capturing temporal patterns. Recently, mixer models based on MLP layers achieve state-of-the-art results through explicitly decomposing and mixing trend and seasonality components in a hierarchical manner, considering both bottom-up and top-down mixing across multiple temporal scales (Wang *et al.*, 2024b). By integrating multi-scale information and capturing intricate temporal dependencies with MLP layers, TimeMixer (Wang *et al.*, 2024b) has shown remarkable success. Recently, state space Models have attracted the attention of researchers due to their sequential modeling power and interpretability. One prominent model leveraging SSM is the Mamba model (Gu et Dao, 2023), which employs a hierarchical Bayesian approach to represent intricate temporal correlations and learn dynamic patterns over time (Gu et Dao, 2023). More recently, the Mamba model has been utilized to capture cross-channel dependencies in time series and has achieved promising results (Wang *et al.*, 2025). However, the potential of RNN models has been overshadowed in time series forecasting domain. Considering the capability of RNNs in sequence modeling, we are interested in exploring the performance of RNNs, specifically GRUs, in multivariate time series forecasting of dairy cattle. In our model, bidirectional GRU modules are employed to capture cross-channel correlations across time series channels, effectively modeling dependencies along both forward and backward directions. The inverted GRU module is integrated with feed-forward layers to represent temporal dependencies, enhancing the model's ability to process time series data.

8.3 Methodology

8.3.1 Problem Definition

The problem of milk production forecasting can be defined as follows : given a herd of cows of size N and each cow i represented by a set of records $\{\mathbf{x}_j^{(i)} = (x_{j,1}^{(i)}, ..., x_{j,T}^{(i)})\}_{j=1..v} \epsilon \Re^{v \times T}$, where $\mathbf{x}_j$ is a sequence

of dairy variables, v representing the number of dairy variables considered and T, the record length (total length of the early lactation periods, fixed for all animals), we aim to predict M steps (months) of the upcoming milk production across the next lactation period : $\hat{r}^{(i)} \, \epsilon \, \Re^M$.

8.3.2 RNN and GRU

Recurrent Neural Networks (RNNs) are a specific type of neural network designed for sequential data processing. Unlike traditional feed-forward neural networks, RNNs incorporate loops that allow them to retain a memory of previous inputs by passing information from one step of the sequence to the next, which makes them particularly well suited for tasks involving time series or other sequential data. However, standard RNNs can suffer from issues like vanishing or exploding gradients, which affect their ability to capture long-term relationships. To address these limitations, advanced RNNs such as Gated Recurrent Units (GRUs) (Cho *et al.*, 2014) and Long-Short Term Memory (LSTM) (Hochreiter et Schmidhuber, 1997a) variants were developed, introducing update and reset gates to selectively control information flow. In our model (bi-iGRU), we leverage GRUs to effectively capture cross-channel dependencies across time series data, enabling robust representation of intricate interactions between time series channels. The Gated Recurrent Unit (GRU) (Cho *et al.*, 2014), introduced in 2014, is a well-known type of RNN with a gating mechanism to input or forget information along a sequence of time steps. It utilizes update and reset units to process sequential data in a hidden space. The processing steps of GRU is represented by the following equations :

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{8.1}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{8.2}$$

$$\hat{h}_t = \phi(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{8.3}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \tag{8.4}$$

Where, $\odot$ represents element-wise multiplication. x_t and h_t denote input and hidden state vectors at time t, respectively. z_t, r_t, $\hat{h}_t$ represent the update, reset, and candidate hidden state at time t, respectively. In addition, W_h, W_z, W_r, U_h, U_z, U_r, b_h, b_z and b_r indicate the associated weights and biases, which are learned in an end-to-end manner. GRU has shown promising sequential modeling in various applications while consuming fewer parameters than LSTM (Wang *et al.*, 2021a), (Lawi *et al.*, 2022). In most applications, including time series forecasting, GRUs and LSTMs are commonly used to capture temporal dependencies (Lin *et al.*, 2023). However, the ability of GRUs to effectively capture cross-channel dependencies in time series data has often been neglected.

8.3.3 Bi-directional Cross-channel GRU for time series forecasting

Time series forecasting using the bi-directional inverted GRU model is depicted in Figure 8.1 (in more detail). Bi-iGRU receives an inverted version of the time series (Liu *et al.*, 2023). In other words, the input series is first inverted by exchanging its time and channel dimensions. The inverted series is then projected to the model space using a linear mapping layer. RNN models usually process a sequence in one direction, and sometimes sequence processing in one direction does not provide enough insight into the sequential dependencies. Accordingly, in bi-directional iGRU architecture, the inverted series and its reversed version are passed through the GRU blocks, separately, to obtain the bi-directional relations between the time series channels (variates) effectively. We utilize separate GRU blocks in each layer in order to process both

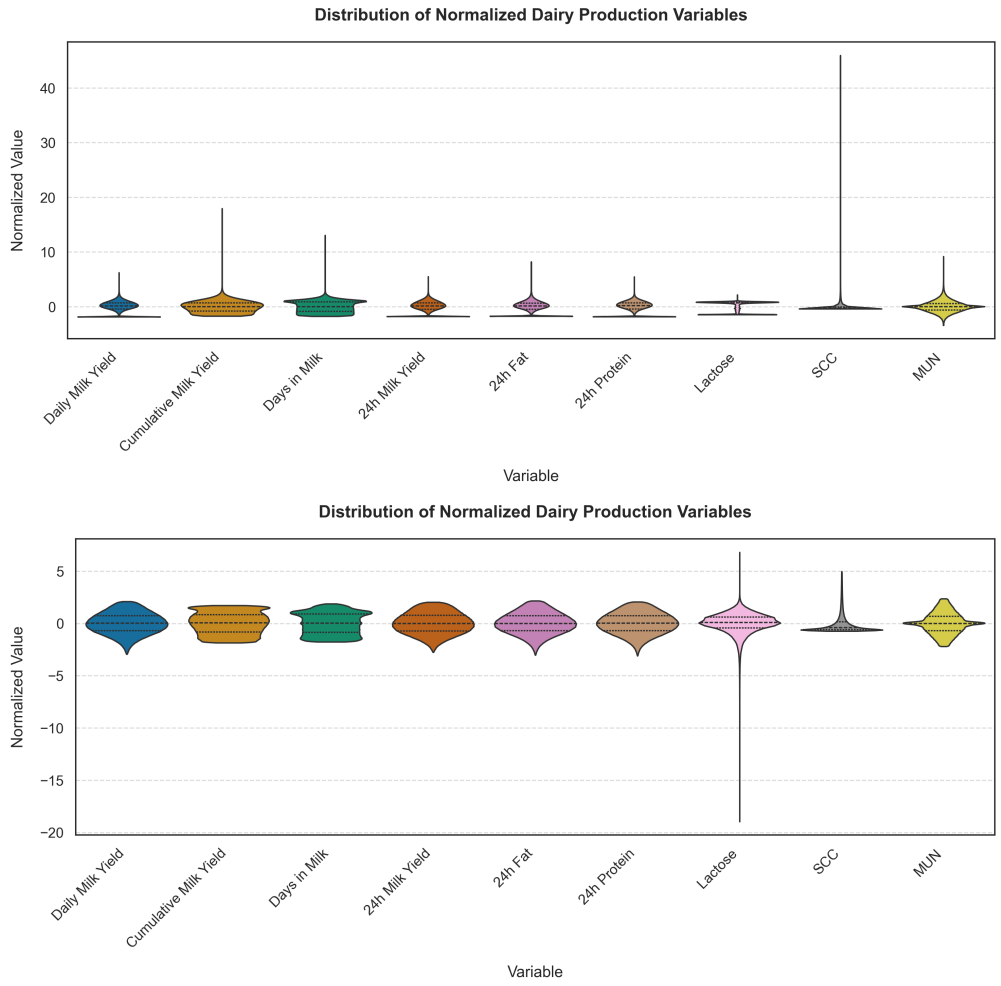
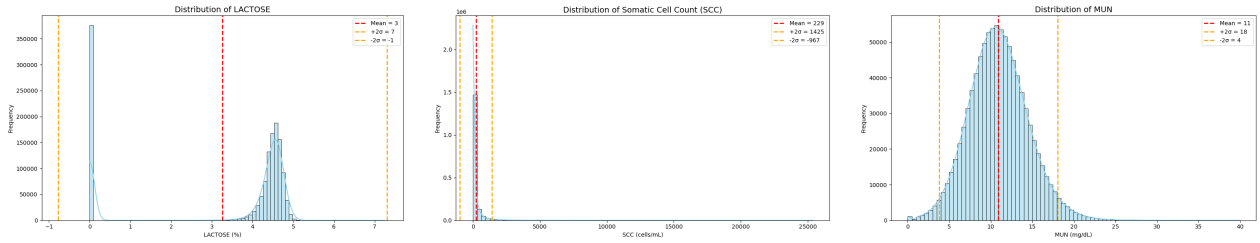


Figure 8.2 – Violin plots of the selected normalized dairy variables : (top) before and (bottom) after the outlier removal step.

the input series and its reversed form. After processing by the corresponding GRU unit, the reversed series is restored to its original order and combined with the processed series through summation. The result is added with the input series using a skip connection and then layer-normalized to stabilize training. We employ a feed-forward layer, which implicitly captures temporal correlations along the model dimension. The feed-forward layer is composed of two linear layers : the first projects the time series from the model dimension to a higher-dimensional feed-forward space (2 times model dimension), followed by a GELU activation function to introduce non-linearity, and the second maps it back to the original model dimension. This enables efficient processing of temporal features while maintaining the representational capacity of the model. The output of the feed-forward layer is combined with its input via a skip connection and then passed to a normalization layer. The final prediction is achieved by feeding the result of the latest step to a projection layer, which is a fully-connected layer that maps the model dimension to the forecasting horizon.

8.4 Dataset Description

Lactanet, a leading Canadian dairy herd management organization, collects milk production data through routine milk recording, bulk tank sampling, and specialized analyses, adhering to International Committee for Animal Recording (ICAR) guidelines. Our dairy dataset is curated and recorded by the Lactanet organization from a large collection of milk production tests, which were conducted once per month, yielding approximately 2154797 records over six lactation periods of dairy cows, after pre-processing steps. The dataset comprises milk production test records from N=29136 dairy cows across 5019 herds in Quebec, Canada, collected between the years 2006 and 2017. Each test record consists of various features obtained from milk samples and management-related measurements. We pre-processed the dataset by imputing missing values, replacing outliers with meaningful values, and retaining only the records of cows that survived until the end of the sixth lactation. For identifying outliers, we determined values beyond 2 standard deviations (from the mean) as outliers for each variable and marked those extreme values as missing data, then treated them in the missing data imputation steps, replacing them with herd-wise and season-wise averages. To verify that this approach did not distort the data or remove biologically meaningful information, we examined the empirical distributions of the key heavy-tailed variables, including Somatic Cell Count (SCC), Lactose and Milk Urea Nitrogen (MUN). As shown in Figure 8.3, more than 97% of the observations fall within the mean $\pm 2\sigma$ range, confirming that the vast majority of biologically plausible records are retained. The remaining extreme values represent highly improbable or erroneous entries. Robust statistical methods (median absolute deviation) were also tried, but we noticed that the critical SCC values were removed by this method. Therefore, we kept the current approach (mean $\pm 2\sigma$) for outlier treatment.



(a) Lactose (%) (b) Somatic Cell Count (SCC, cells/mL) (c) Milk Urea Nitrogen (MUN, mg/dL)

Figure 8.3 – Empirical distributions of key milk composition and health indicators. The plots show (a) Lactose, (b) Somatic Cell Count (SCC), and (c) Milk Urea Nitrogen (MUN), along with their mean (red dashed line) and $\pm 2\sigma$ thresholds (orange dashed lines). More than 97% of the data fall within the $\pm 2\sigma$ range, highlighting that biologically plausible records were preserved while extreme outliers were filtered.

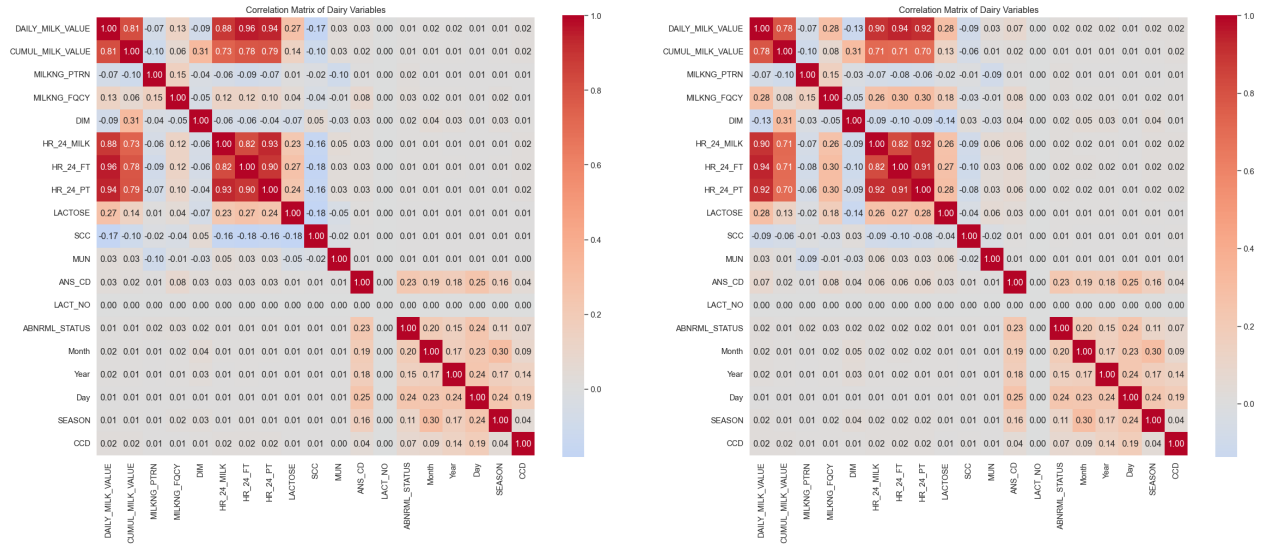
To ensure consistency and completeness, duplicate records were deleted, and inconsistent records were imputed with zeros to maintain dataset size, as removal would reduce records. For instance, the records with negative milk income or negative cumulative milk income were imputed with a default value of zero. Only 0.001% of the records had negative daily milk income or negative cumulative milk income before the replacement phase. Negative daily milk income values were primarily due to data entry errors, while negative cumulative income reflected rare inconsistencies in long-term records. These records were imputed with zeros to ensure data integrity, as negative values for income are biologically implausible for milking cows. Additionally, we removed records associated with the dry period of animals during their lactation periods. For missing data imputation in the dataset, we employed a three-step approach. First, we grouped animals

by herd ID, test year, and season, imputing missing values within each group using the mean of each group to which the animal belongs. Next, for any remaining missing values, we imputed them using the mean of the associated herd, based on herd ID. Finally, we addressed any still-unresolved missing values by imputing them with the group mean defined by the test year. The aforementioned three-step imputation approach leverages the hierarchical structure of the dairy dataset to guarantee accurate and contextually relevant replacements for missing values. By first grouping animals by herd ID, test year, and season, local patterns specific to environmental and management conditions are captured, using group means to impute missing information. The subsequent step, imputing with herd-level means, considers broader herd-specific trends, preserving consistency and similarity across animals within the same herd. Finally, using test year averages addresses the remaining gaps with a reasonable replacement based on annual trends. These steps minimize bias by prioritizing the most specific available data while progressively considering broader replacements. At the end, 19 relevant dairy variables were selected and used to construct a multivariate time series based on their recorded variable sequences over six lactation periods (72 months). The selected dairy variables include, daily milk income, cumulative milk income, milking pattern, milking frequency, ANS-CD (animal status code), lactation number, days in milk, milk produced in 24 hours (kg), 24-hour fat, 24-hour protein, lactose, abnormal status, Somatic cell count (SCC), Milk Urea Nitrogen (MUN), test year, test month, test day, test season, and condition code (a sickness indicator). The description of the selected dairy variables is shown in Table 8.1. In addition, we converted the categorical variables into one-hot encoded versions in order to leverage their corresponding information, more efficiently. The continuous variables were normalized using mean and standard deviation. The distribution of normalized continuous dairy variables before and after outlier treatment is illustrated in Figures 8.2. According to Figure 8.2 (top), most variables exhibit outlier values in both upper and lower tails, with the SCC factor showing the most extreme values, reaching up to 45. As shown in Figure 8.2 (bottom), most variables follow a more symmetric distribution after outlier replacement phase, though lactose retains some extreme values around -20. We further validate our outlier treatment approach through correlation calculation between the dairy variables before and after outlier replacement step. According to Figure 8.4, the correlation patterns and similarities remained almost the same, with some correlation scores improved after this step. Post-imputation correlations (Figure 8.4 left) show cumulative milk income strongly correlated with fat yield (HR 24 FT) (r = 0.78, vs. 0.71 pre-imputation), consistent with Lactanet's pricing models, and SCC inversely correlated with both daily and cumulative milk income (r = -0.17, -0.10), reflecting udder health impacts. The comparison between correlation scores with and without outliers, highlights that the outlier treatment retains the essential correlations, while removing extreme and unrelated feature values. Finally, we obtained 97 features associated with the time series channels. In our initial assessment of the dairy dataset, we examined the correlations between variables such as milk production, health indicators, milk quality, and seasonality factors. Figure 8.4 supports our hypothesis that cross-channel dependencies are critical for effective time series forecasting of dairy income. For example, daily milk value (income) and cumulative milk income are strongly correlated with 24-hour milk, fat, and protein production. Therefore, capturing these critical interactions helps achieve accurate results. We illustrated different steps of our proposed pipeline, including data pre-processing, training, and testing steps, in Figure 8.5.

The dataset is randomly split into train and test parts with 80 :20 ratio. We created six different versions of the dataset, each corresponding to a different lactation period treated as the forecast horizon. In each case, the input consists of records from one or more previous lactation periods, while the output represents the upcoming lactation period. For instance, in the first version, only the first lactation is used as input, and the second lactation serves as the forecast horizon. In the second version, the input includes data from the first and second lactation periods (covering 24 months), while the third lactation (months 25 to 36) is the forecast horizon. The other four versions follow this pattern, progressively incorporating more lactation

Table 8.1 – Description of dairy variables used to forecast milk income.

Variable	Description	Min	Max
Daily Milk Value	Income from milk produced in the test day	0	74.59
Cumulative Milk Value	Cumulative milk income along the lactation months	0	49477
Milking Pattern	Milking pattern of an animal	1	4
Milking Frequency	The milking frequency of the animal for the test (1 time per day, 2 times per day)	0	3
Animal Status Code	Animal status indicator (1 indicates dryness, 2 indicates milking)	1	4
Lactation Number	Lactation number in {1, 2, 3, 4, 5, 6}	1	6
Days in Milk	Number of days that a cow has been milking along the lactation months	0	1575
24-hour-Milk	Milk produced (Kg) in 24 hours	0	90.9
24-hour-Fat	Fat in milk produced in 24 hours	0	5.802
24-hour-Protein	Protein in milk produced in 24 hours	0	3.342
Lactose	Lactose percentage in milk sample (24 hours)	0	7.25
Abnormal Status	Quality of the sampled data on the test day	0	8
Somatic Cell Count (SCC)	Somatic cell count in the milk sample	1	25339
Milk Urea Nitrogen (MUN)	Milk Urea Nitrogen (mg / dL)	0	39.9
Test Year	Year indicator of the test date	6	17
Test Month	Month of the test date	1	12
Test Day	Day of the test date	1	31
Test Season	Season of the test	1	4
Condition Code	Code indicator of the animal condition (0 healthy, 1 sick)	0	1



(a) Without outliers

(b) With outliers

Figure 8.4 – An illustration of the correlations between different dairy variables, with and without outliers.

periods as input and the next lactation as the forecast window (12 months). We train time series forecasting models with those dairy datasets. After training the models, we have a collection of models to forecast the dairy income along various lactation periods, including lactation 2, 3, 4, 5 and 6. Therefore, we can provide suggestions regarding future profitability to dairy farmers in different lactation periods.

8.5 Results and Discussion

The bidirectional iGRU model is compared with carefully selected state-of-the-art models, including TimeMixer++ (Wang *et al.*, 2024a), S-Mamba (Wang *et al.*, 2025), TimeMixer (Wang *et al.*, 2024b), iTransformer (Liu *et al.*, 2023), PatchTST (Nie *et al.*, 2022), DLinear (Zeng *et al.*, 2023), TimesNet (Wu *et al.*, 2022), MuMu+Attention (Naghashi *et al.*, 2023), and SegRNN (Lin *et al.*, 2023). The proposed model is implemented in Pytorch (Paszke *et al.*, 2019) and the experiments were conducted using a single A100-40G NVIDIA GPU. We train our model and other baselines for 15 epochs using Mean Square Error (MSE) loss and the Adam optimizer (Kingma et Ba, 2014). We performed hyper-parameter tuning by trying a range of different values, and selecting the value which yielded the best result. In our experiments, the batch size was uniformly selected as 32, and the number of encoder blocks and model dimension were set to 3 and 512, respectively. Two common error metrics were selected to evaluate the models. We assess models both in the forecasting of cumulative milk income and the prediction of the whole dairy channels (97 features) :

$$\text{MSE}_{\text{milk}} = \frac{1}{n \cdot T} \sum_{i=1}^{n} \sum_{t=1}^{T} (y_{i,t} - \hat{y}_{i,t})^2 \tag{8.5}$$

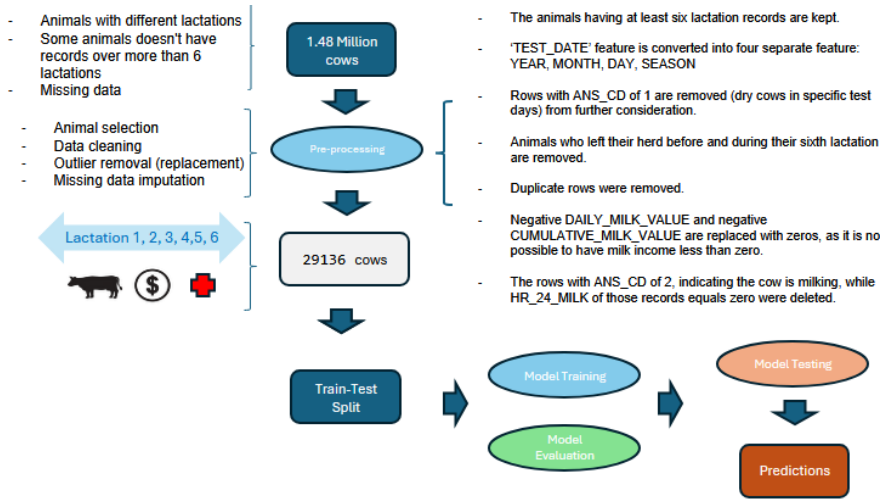


Figure 8.5 – An illustration of different steps in time series forecasting of dairy production : Pre-processing, model training and testing phases.

$$\text{MAE}_{\text{milk}} = \frac{1}{n \cdot T} \sum_{i=1}^{n} \sum_{t=1}^{T} |y_{i,t} - \hat{y}_{i,t}| \tag{8.6}$$

$$\text{MSE}_{\text{total}} = \frac{1}{n \cdot F \cdot T} \sum_{i=1}^{n} \sum_{f=1}^{F} \sum_{t=1}^{T} (x_{i,f,t} - \hat{x}_{i,f,t})^2 \tag{8.7}$$

$$\text{MAE}_{\text{total}} = \frac{1}{n \cdot F \cdot T} \sum_{i=1}^{n} \sum_{f=1}^{F} \sum_{t=1}^{T} |x_{i,f,t} - \hat{x}_{i,f,t}| \tag{8.8}$$

Where, n indicates the number of cows, T is the forecast horizon set to 12, F is the number of dairy variables (channels), $y_{i,t}$ represents the actual cumulative milk income for cow i at month t, $\hat{y}_{i,t}$ denotes the predicted cumulative milk income for cow i at month t, $x_{i,f,t}$ is the actual value of feature f for cow i at month t, and $\hat{x}_{i,f,t}$ indicates the predicted value of feature f for cow i at month t.

The forecast results for cumulative milk income and total variates are reported in Tables 8.2 and 8.3. Tables 8.4 and 8.5 show the average of different error metrics across the lactation periods : 2, 3, 4, 5, and 6. According to these tables, Bi-iGRU delivers competitive results in most cases, specifically it achieves better results compared to recently proposed TimeMixer++, TimeMixer, and iTransformer. It also attains competitive or better results relative to S-Mamba and MuMu+Attention models. Bi-iGRU demonstrates a 6% improvement in MSE compared to the TimeMixer++ model when predicting the cumulative production of the fifth lactation period, while requiring less training time. It also outperforms S-Mamba by more than 8% in terms of MSE reduction in the prediction of the sixth lactation period. Bi-iGRU reduces the MSE error by more than 46% compared to the iTransformer model in the prediction of the cumulative income of the sixth lactation.

Table 8.2 – Forecasting results of total dairy variates across models and lactation periods.

Input - Target Lactation	**1 - 2**		**1, 2 - 3**		**1, 2, 3 - 4**		**1, 2, 3, 4 - 5**		**1, 2, 3, 4, 5 - 6**	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Bi-iGRU	**0.069**	**0.106**	**0.072**	**0.109**	**0.076**	**0.113**	**0.079**	0.115	**0.086**	0.122
TimeMixer++	0.091	0.112	0.083	0.112	0.085	0.115	0.089	0.119	0.093	0.121
SegRNN	0.088	0.128	0.083	0.124	0.086	0.127	0.089	0.129	0.101	0.136
TimeMixer	0.084	0.136	0.078	0.126	0.080	0.125	0.084	0.127	0.089	0.129
iTransformer	0.125	0.148	0.101	0.132	0.101	0.139	0.105	0.139	0.117	0.150
S-Mamba	0.083	0.124	0.083	0.124	0.089	0.129	0.079	**0.108**	0.087	**0.118**
TimesNet	0.205	0.273	0.114	0.198	0.118	0.209	0.116	0.203	0.115	0.184
DLinear	0.073	0.116	0.075	0.116	0.078	0.117	0.082	0.120	0.087	0.124
PatchTST	0.135	0.162	0.105	0.141	0.103	0.138	0.105	0.138	0.117	0.150

It also demonstrates a 45% reduction in the MSE error compared to the SegRNN model, which is based on an RNN architecture and time series segments. This highlights the superior performance of Bi-iGRU over other RNN-based models. Bi-iGRU excels over TimeMixer++ in forecasting different lactation periods, except the sixth lactation. In particular, it outperforms TimeMixer++ in predicting cumulative income during the second lactation period by more than 34%, and achieves an improvement of 24% in the prediction of total dairy variates, highlighting the efficiency of GRU in capturing cross-channel correlations in forward and backward directions. Bi-iGRU also demonstrates promising performance compared to TimeMixer++ when forecasting the total variates across various lactation periods. Specifically, it achieves improvements of more than 13%, 10.5%, and 11% throughout the third, fourth, and fifth lactation periods, respectively (Table 8.2). However, Bi-iGRU achieves slightly better results than the MuMu+Attention model with overall MSE improvement of 2% averaged over the lactation periods (Table 8.4). These results highlight the model's robustness across multiple stages of lactation and its ability to generalize well over extended time horizons (from early lactation to late lactation periods). Consistent gains indicate that Bi-iGRU is an effective model for capturing the underlying temporal and cross-channel dependencies necessary for accurate forecasting in complex real-world dairy datasets.

8.5.1 Dairy forecasting with different input length

Table 8.6 presents the performance of Bi-iGRU when forecasting milk income across lactation periods 2 to 6, using varying numbers of past lactation periods as context. Here, a clear and prominent trend emerges : The accuracy of the model improves as more historical lactation data is provided. For instance, when predicting the cumulative income during the sixth lactation period, Bi-iGRU achieves an MSE of 0.258 using all five prior lactation records (1-5) as input, compared to an MSE of 0.408, when using only the fifth lactation (a 36.76% improvement). This MSE reduction can be attributed to the bidirectional cross-channel GRU architecture, which effectively leverages the additional context from past lactation periods to capture long-term temporal and cross-channel correlations. For example, variables like cumulative milk income and 24-hour fat yield, which are strongly correlated (r=0.78 post-imputation, as shown in Figure 8.4), benefit from a longer input horizon that allows the model to better understand their evolving relationship over time. From a practical

Table 8.3 – Forecasting results of cumulative milk income across models and lactation periods.

Input - Target Lactation	**1 - 2**		**1, 2 - 3**		**1, 2, 3 - 4**		**1, 2, 3, 4 - 5**		**1, 2, 3, 4, 5 - 6**	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Bi-iGRU	**0.134**	**0.260**	**0.134**	**0.258**	**0.144**	**0.264**	0.141	**0.265**	0.227	0.327
TimeMixer++	0.206	0.316	0.150	0.277	0.155	0.277	0.150	0.277	**0.213**	**0.316**
SegRNN	0.196	0.315	0.148	0.277	0.155	0.278	0.149	0.278	0.414	0.442
TimeMixer	0.158	0.291	0.144	0.273	0.150	0.273	0.146	0.275	0.227	0.332
iTransformer	0.242	0.344	0.183	0.306	0.184	0.297	0.195	0.431	0.419	0.431
S-Mamba	0.148	0.277	0.148	0.277	0.149	0.278	**0.140**	0.266	0.229	0.329
TimesNet	0.365	0.422	0.200	0.338	0.209	0.340	0.193	0.328	0.256	0.366
DLinear	0.138	0.270	0.139	0.268	0.144	0.268	0.142	0.272	0.218	0.325
PatchTST	0.325	0.430	0.205	0.343	0.181	0.318	0.180	0.326	0.436	0.493

Table 8.4 – Average model performance metrics for total variates across lactations 2 to 6.

Model	**MSE**	**MAE**	**RMSE**	**R^2**
MuMu+Attention	0.0776	0.1190	0.2785	0.5877
Bi-iGRU	**0.0759**	0.1119	**0.2753**	**0.5970**
PatchTST	0.1154	0.1465	0.3394	0.3943
TimesNet	0.1310	0.2086	0.3590	0.2953
TimeMixer++	0.0881	0.1194	0.2966	0.5306
S-Mamba	0.0761	**0.1093**	0.2756	0.5928
SegRNN	0.0890	0.1281	0.2981	0.5271
iTransformer	0.1127	0.1431	0.3323	0.4100
TimeMixer	0.1073	0.1628	0.3275	0.4281
DLinear	0.1176	0.1814	0.3428	0.3733

perspective, this suggests that dairy farmers can achieve more accurate forecasts of future profitability by maintaining comprehensive records over multiple lactation periods. However, the marginal improvement decreases as more lactation records are added (e.g. 17.31% improvement when using 5 prior lactations compared to 4 recent lactation periods), indicating a potential saturation point where additional historical data might yield repetitive temporal patterns.

Table 8.5 – Average model performance metrics for cumulative milk income across lactations 2 to 6

Model	MSE	MAE	RMSE	R^2
MuMu+Attention	**0.1516**	0.2722	**0.3878**	**0.3821**
Bi-iGRU	0.1518	**0.2712**	0.3880	0.3818
PatchTST	0.2687	0.3867	0.5107	-0.0553
TimesNet	0.2430	0.3571	0.4890	-0.1214
TimeMixer++	0.1889	0.3045	0.4317	0.2073
S-Mamba	0.1550	0.2744	0.3915	0.3712
SegRNN	0.2106	0.3162	0.4481	0.2024
iTransformer	0.2380	0.3306	0.4789	0.0680
Time Mixer	0.2336	0.3442	0.4740	0.0544
DLinear	0.2851	0.4039	0.5267	-0.1475
ARIMA	2.0810	0.9600	1.1720	-12.1898

8.5.2 Interpretability

We analyzed the correlations between predicted cumulative milk income and each feature from the historical time series after training. This post-hoc analysis reveals the importance and impact of each variable in forecasting cumulative milk income. Figure 8.6 reveals strong positive dependencies with recent production indicators such as cumulative and daily milk value, fat, and protein content, pinpointing that Bi-iGRU effectively prioritizes relevant traits. On the other hand, negative correlations with abnormal status, specific seasonal and year variables suggest that the model captures health and environment-related effects on income. Overall, these patterns align with biological expectations and highlight that forecasts are both data-driven and interpretable.

8.5.3 Statistical significance

To ensure statistical rigor and account for multiple hypothesis testing, pairwise independent t-tests were performed between Bi-iGRU model and other competitive baselines, including iTransformer, S-Mamba, TimeMixer, TimeMixer++, and SegRNN. Figure 8.7 represents the box plots for MSE and MAE comparisons across these models. To prevent Type I error due to multiple comparisons, the raw p-values were adjusted according to the Benjamini–Hochberg false discovery rate (FDR) correction method. In terms of MSE, Bi-iGRU outperforms most competing models during the fifth lactation period, particularly TimeMixer++ ($t = -16.7332$, adjusted $p < 0.001$), indicating a robust and statistically significant reduction in prediction error. The comparison with S-Mamba, however, shows a nonsignificant difference ($t = 0.4752$, adjusted $p = 0.6420$), indicating comparable performance. For MAE, Bi-iGRU also demonstrates relatively better performance than SegRNN, iTransformer, TimeMixer, and TimeMixer++ (adjusted $p < 0.001$ for all). Although Bi-iGRU slightly outperforms S-Mamba, the difference remains non-significant after correction (adjusted $p = 0.065$). These results confirm that Bi-iGRU achieves statistically significant improvements in both MSE

Table 8.6 – Bi-iGRU forecasting results for lactations 2–6 with varying input lengths. *Total* corresponds to the total dairy forecasting, and *Cumulative* refers to forecasting of the cumulative milk income.

	Target Lact	**2**	**3**			**4**			**5**				**6**			
	Input Lact(s)	1	2	1,2	3	2,3	1,2,3	4	3,4	2,3,4	1,2,3,4	5	4,5	3,4,5	2,3,4,5	1,2,3,4,5
MSE	Total	0.070	0.074	0.073	0.078	0.077	0.077	0.082	0.081	0.081	0.082	0.097	0.096	0.093	0.092	0.090
	Cumulative	0.135	0.140	0.135	0.153	0.146	0.145	0.149	0.143	0.142	0.141	0.408	0.400	0.365	0.312	0.258
RMSE	Total	0.264	0.272	0.270	0.279	0.277	0.277	0.286	0.285	0.285	0.286	0.312	0.310	0.305	0.303	0.300
	Cumulative	0.367	0.374	0.367	0.391	0.382	0.381	0.386	0.378	0.377	0.376	0.639	0.633	0.604	0.559	0.508
MAE	Total	0.109	0.112	0.112	0.114	0.114	0.115	0.117	0.117	0.118	0.119	0.129	0.129	0.127	0.127	0.126
	Cumulative	0.261	0.264	0.259	0.273	0.266	0.264	0.274	0.268	0.265	0.265	0.435	0.431	0.415	0.379	0.347

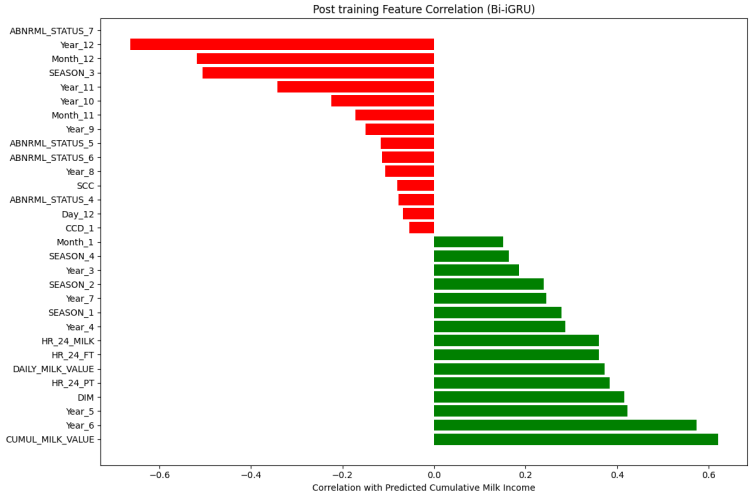
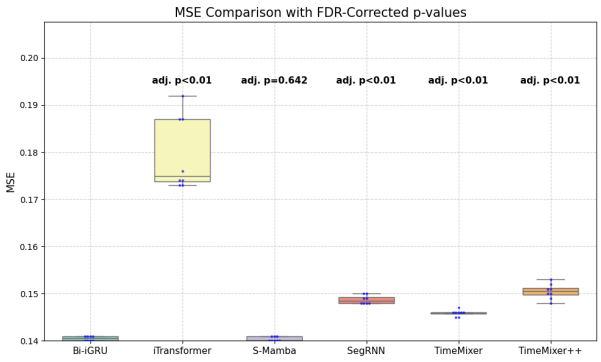


Figure 8.6 – Correlation between the predicted cumulative milk income with the context dairy variables after model training.

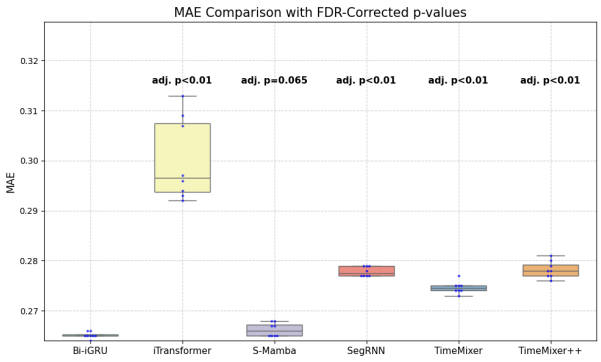
and MAE compared to most baselines, and that these findings remain robust even after applying multiple-testing correction, underscoring the reliability of the results.

8.5.4 Ablation study

The ablation study in Table 8.7 evaluates the impact of removing key components of the Bi-iGRU model, namely the feed-forward layer, the bidirectional mechanism and the GRU modules, on the prediction performance for the fifth and sixth lactation periods. When the feed-forward layer is removed (W/O Feed-Forward), the MSE for cumulative milk income in the sixth lactation increases from 0.242 to 0.327 (a 26% increase), highlighting the critical role of the feed-forward layer in capturing temporal dependencies along the model dimension. This layer, composed of two linear transformations with a GELU activation, introduces non-linearity that boosts the model's ability to capture complex temporal patterns, such as the gradual increase in cumulative milk income over lactation months. Similarly, switching to a unidirectional GRU (Unidirectional case) results in an MSE of 0.402 for the sixth lactation, a 40% increase compared to Bi-iGRU, un-



(a) Comparison of model performance in MSE with statistical tests.



(b) Comparison of model performance in MAE with statistical tests.

Figure 8.7 – Results of the conducted statistical tests and the relevant box plots (Forecasting of the fifth lactation period).

Table 8.7 – Ablation study of Bi-iGRU model by removing the feed-forward layer, GRU modules and two-directional channel-wise GRU.

Design		W/O Feed-Forward		Unidirectional GRU		W/O GRU		Bi-iGRU	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Lactation 6	Total	0.092	0.128	0.095	0.128	0.118	0.168	**0.089**	**0.125**
	Cumulative	0.327	0.393	0.402	0.423	0.430	0.443	**0.242**	**0.338**
Lactation 5	Total	0.082	0.119	0.083	0.119	0.105	0.161	**0.082**	**0.119**
	Cumulative	**0.141**	0.266	0.144	**0.264**	0.173	0.292	0.142	0.265

derscoring the importance of processing the time series in both directions to capture cross-channel relations comprehensively. Finally, removing the GRU modules entirely (W/O GRU) leads to the highest cumulative income MSE (0.430), indicating that the GRU's gating mechanism is essential for selectively retaining and forgetting information across time series channels. These experiments validate the synergistic design of Bi-iGRU, where each component contributes to its overall performance. We further studied hyper-parameter sensitivity of our model by conducting experiments using different values of the key hyper-parameters. Figure 8.8 illustrates the test error under different configurations. The model dimension, learning rate, and number of training epochs were varied and the mean squared error (MSE) was calculated in each setting. The model achieves the best performance at a dimension of 512, learning rate 0.001, and 15 epochs. Bi-iGRU model was trained and tested with different feature subsets, including health, milk quality, and management features on prediction of the cumulative income along the sixth lactation period. We included historical cumulative milk income in all cases. Results (Figure 8.9) show that seasonal features alone produce the best performance (MSE=0.2165, MAE=0.3167, R2=0.4456), followed by the milk quality, management and health subset.

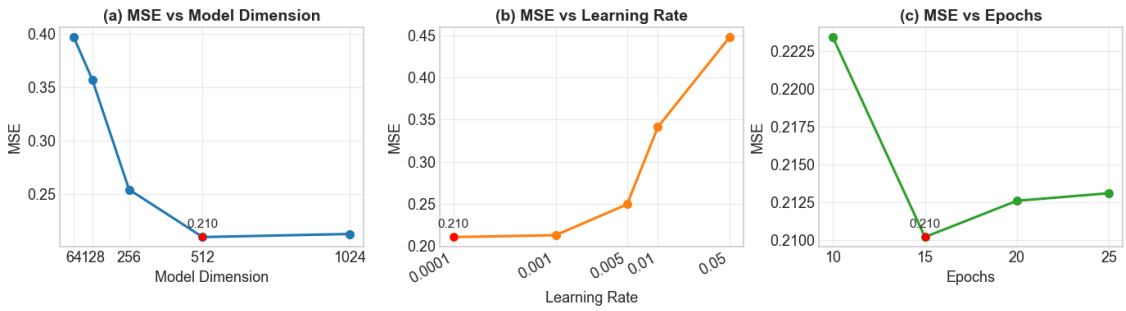


Figure 8.8 – Hyperparameter sensitivity analysis of the proposed model.

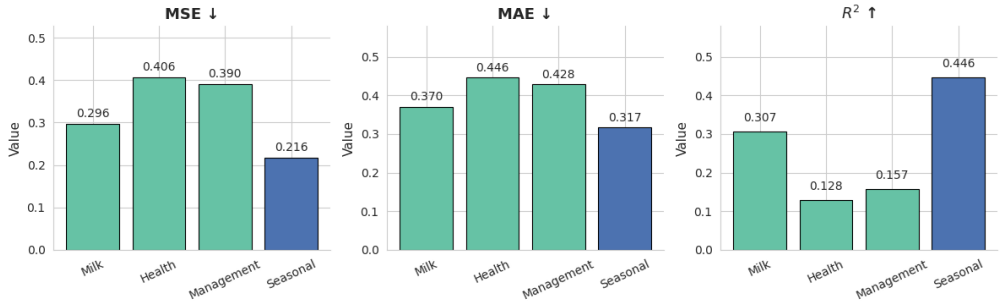


Figure 8.9 – Bi-iGRU performance using different feature subsets on cumulative milk income prediction during the sixth lactation. Seasonal features yield the best overall results.

8.5.5 Visualization

We intuitively demonstrate the superior performance of Bi-iGRU by visualizing the predicted versus ground truth cumulative milk income and daily milk production during different lactation periods. Figures 8.10 to 8.13 provide visual comparisons of Bi-iGRU predictions against baseline models for daily and cumulative milk income in different lactation periods. Figure 8.10 illustrates the prediction of daily milk income during the second lactation, where Bi-iGRU closely follows the ground truth trend, specially during the initial months, where a sharp decrease is observed before stabilization. In contrast, models like TimeMixer and iTransformer underestimate the target values, failing to adapt to the rapid decline. Figure 8.11, which shows cumulative milk income for the fourth lactation period, further highlights Bi-iGRU's accuracy, with predictions aligning closely with the ground truth, while baselines like iTransformer exhibit larger deviations, particularly during the last months, possibly due to their reliance on channel-wise attention mechanisms that might overlook fine-grained dependencies. Figure 8.12, focusing on daily milk income during the fourth lactation, reveals that Bi-iGRU and S-Mamba achieve the most accurate predictions, with Bi-iGRU slightly outperforming S-Mamba in capturing the subtle fluctuations in income, such as the variations in the beginning and end of the lactation period. Finally, Figure 8.13 demonstrates Bi-iGRU's superior performance in forecasting cumulative milk income during the sixth lactation, particularly in the final months, where its delivered predictions remain closely aligned with the ground truth, unlike other models that diverge significantly. These accurate predictions in later lactation months are critical for dairy farmers, as they facilitate reliable long-term planning, such as estimating total income at the end of a lactation cycle to make informed decisions about herd replacement or investment in farm infrastructure.

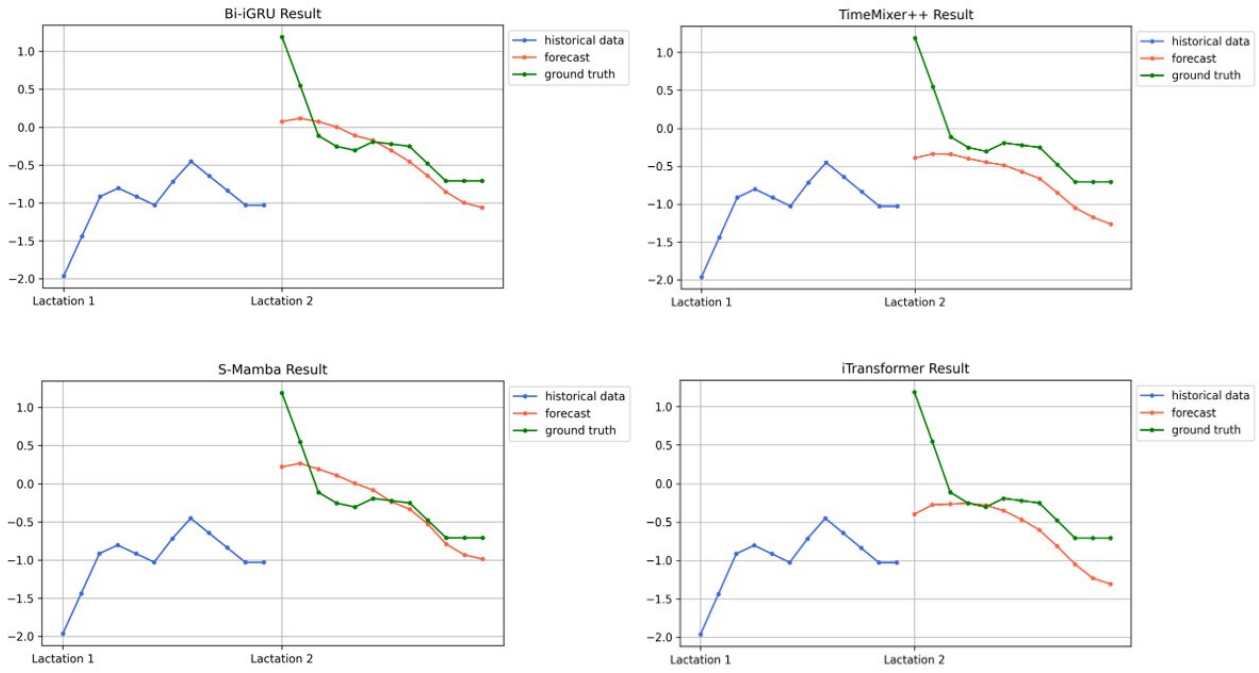


Figure 8.10 – Visual comparison of prediction results among different models for forecasting daily milk income during the second lactation.

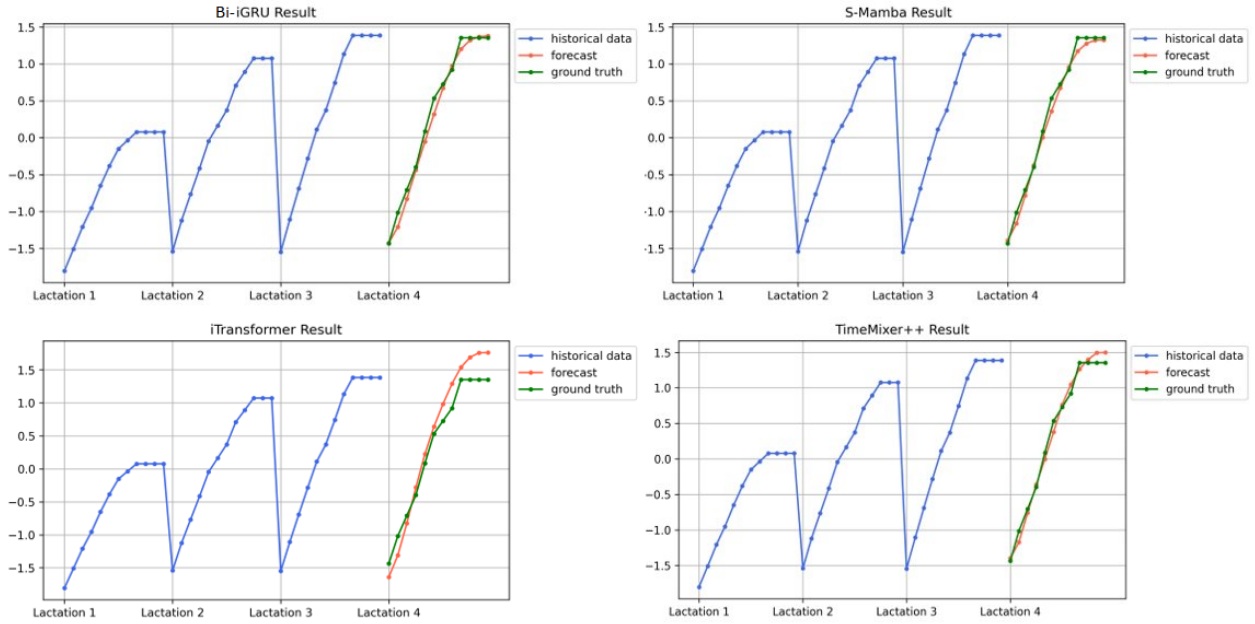


Figure 8.11 – Visual comparison of prediction results among different models for forecasting cumulative milk income during the 4th lactation.

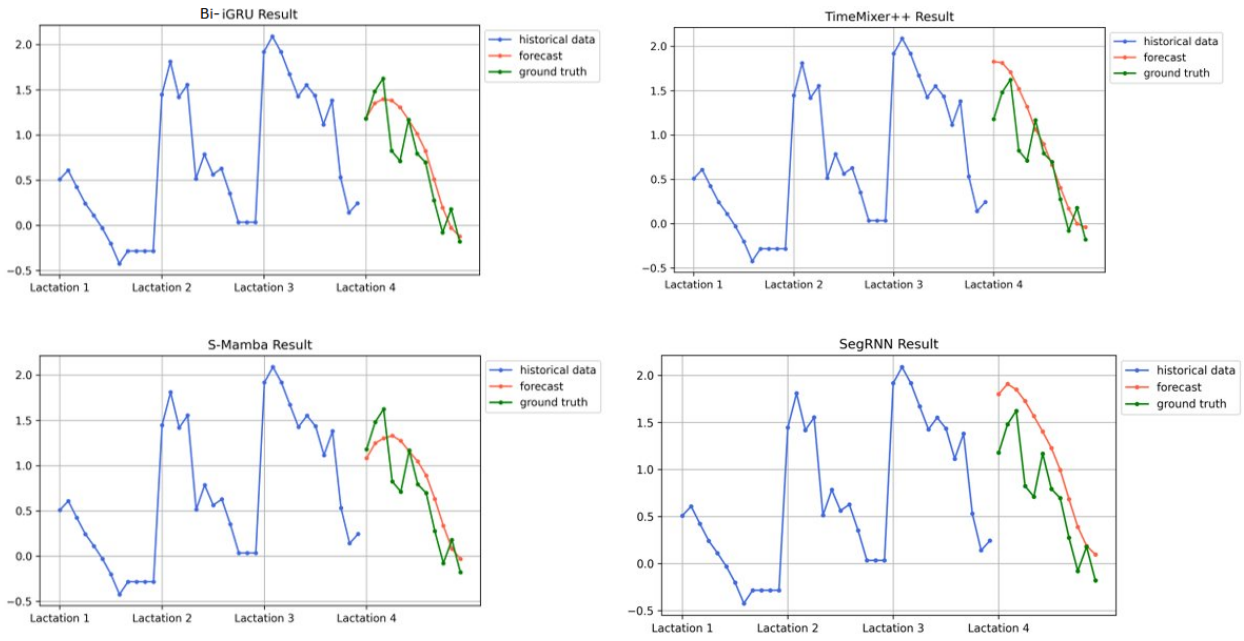


Figure 8.12 – Visual comparison of prediction results among different models for forecasting daily milk income during the 4th lactation.

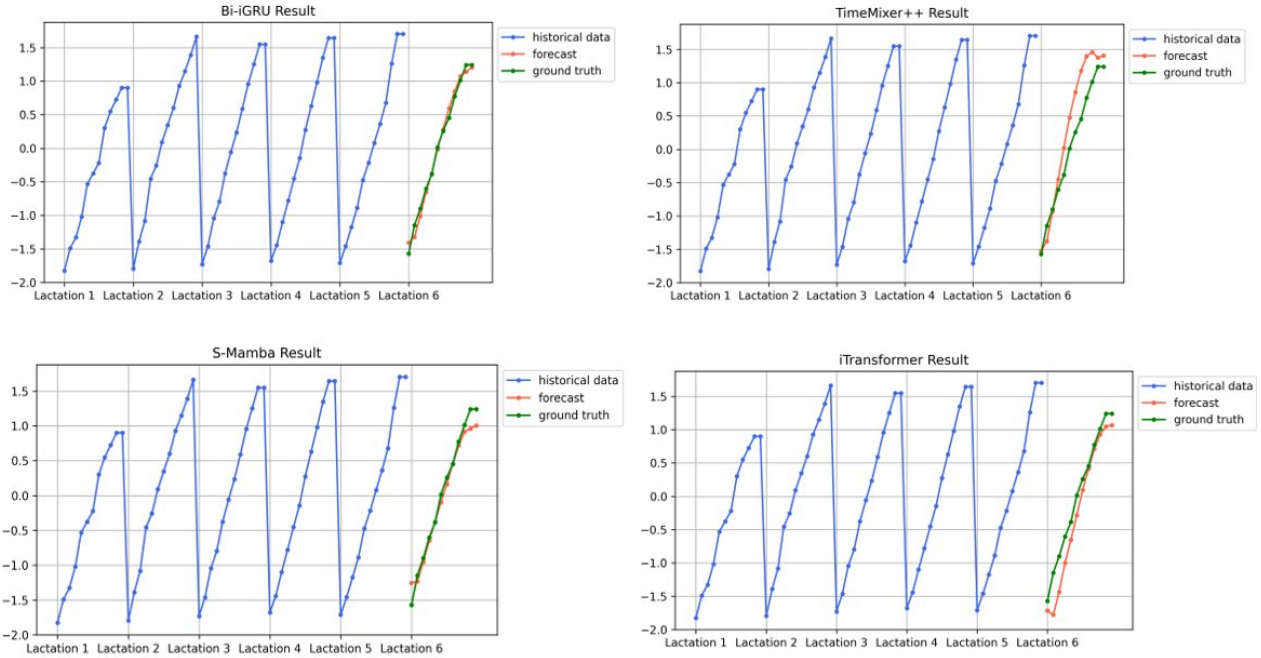


Figure 8.13 – Visual comparison of prediction results among different models for forecasting cumulative milk income during the 6th lactation.

8.5.6 Model efficiency

To verify the computational efficiency of our model, we compare the memory consumption and training time with those of the other baselines. Independent runs were performed using a single A100-40G GPU with the batch size fixed at 32. The efficiency of our model is illustrated in Figure 8.14, where bubble charts display a visual comparison of efficiency metrics. In this Figure, the vertical axis indicates the MAE of the predicted cumulative milk income, and the horizontal axis shows the duration of one training iteration. The bubble size represents the peak memory footprint in one epoch (in gigabytes). As Figure 8.14 illustrates, Bi-iGRU sacrifices a small amount of memory and time to achieve a competitive improvement in accuracy, with the lowest MAE (0.265). On the other hand, DLinear is the fastest model, with the shortest train time (3.05 ms/iter), while achieving relatively low accuracy among the other models. SegRNN, S-Mamba, TimesNet, TimeMixer, and TimeMixer++ show higher error, with TimeMixer++ being slower than others while consuming less memory. PatchTST demands high memory and training time compared to the other models. iTransformer requires similar memory and train time as Bi-iGRU, while delivering the worst result (MAE of 0.431). In addition, we report parameter count, FLOPs and training time for each model in Table 8.8. Although Bi-iGRU has the largest number of parameters, its recurrent architecture allows for parameter reuse across time steps, resulting in lower FLOPs and faster training compared to models with fewer parameters but more computationally intensive operations, such as self-attention or 2-dimensional convolutions. Overall, Bi-iGRU offers an optimal balance of accuracy and efficiency, with S-Mamba achieving the second-best performance.

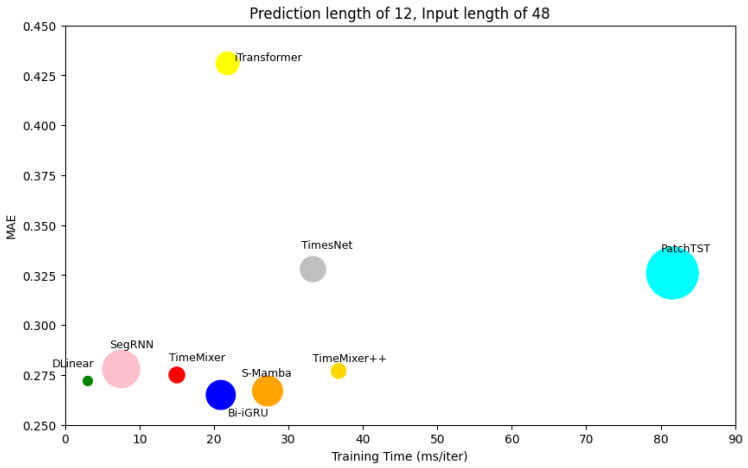


Figure 8.14 – Comparison of model efficiency in terms of peak GPU memory and training time.

8.5.7 Model stability

Figure 8.15 illustrates the distribution of MSE and MAE across independent runs for Bi-iGRU and baseline models, forecasting the fifth lactation period. Bi-iGRU's violin plot (highlighted in blue) depicts a narrow distribution with a mean MSE of 0.141 and a low variance, indicating high stability and consistency in its predictions. The internal quartiles reveal that 75% of Bi-iGRU's MSE values fall below 0.145, a threshold that iTransformer (mean MSE : 0.195) and SegRNN (mean MSE : 0.149) fail to achieve, as their distributions are wider and shifted towards higher errors. This stability is likely due to the layer normalization and skip connections in Bi-iGRU's architecture (Section 3.3), which stabilize training. For dairy farmers, this consistency across various runs translates to reliable forecasts that can be trusted for critical decisions, such as

Table 8.8 – Comparison of models : Parameters, FLOPs, and training time

Model	Params	FLOPs	Training time (ms/iter)
Bi-iGRU	11.07 Million	1076.81 MFLOPs	20.92 ms
MuMu+Attention	4.212 Million	174.52 MFLOPs	14.70 ms
S-Mamba	6.86 Million	1350 MFLOPs	27.20 ms
PatchTST	1.62 Million	3740 MFLOPs	81.58 ms
SegRNN	1.2 Million	926 MFLOPs	7.54 ms
TimeMixer	0.07 Million	566.17 MFLOPs	15.02 ms
DLinear	0.0012 Million	0.12 MFLOPs	3.06 ms
iTransformer	4.757 Million	461.18 MFLOPs	21.82 ms
TimeMixer++	1.18 Million	18538.53 MFLOPs	36.75 ms
TimesNet	3.53 Million	633.33 MFLOPs	33.30 ms

adjusting milking frequency or monitoring SCC to prevent mastitis (a health issue reflected by high SCC values). The narrow error distribution also underscores that Bi-iGRU is less sensitive to variations in the dataset, such as those introduced by the outlier treatment phase, making it a robust choice for real-world applications where data quality may vary.

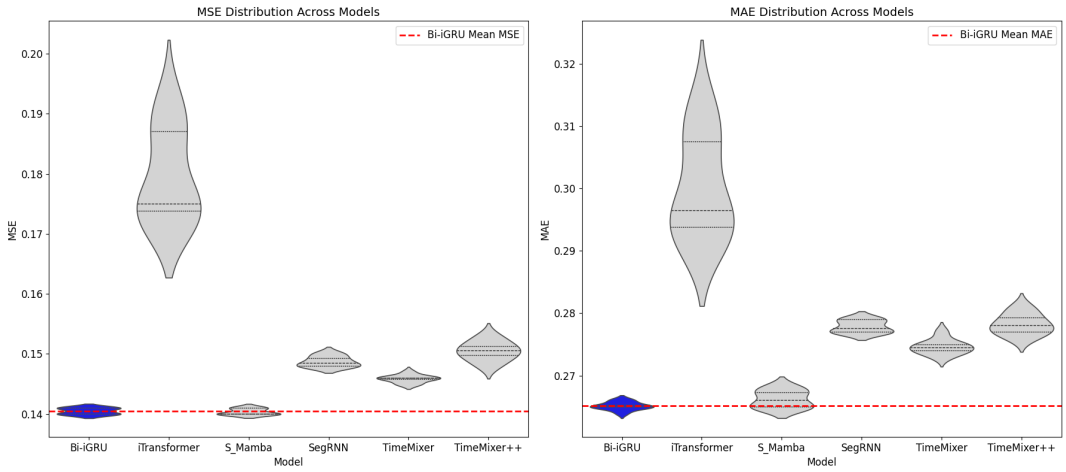


Figure 8.15 – Violin plots for illustration of robustness of our model (Forecasting of the 5th lactation period.)

8.6 Economic and decision-making interpretability

Bi-iGRU achieved a competitive error (MAE of 0.260), when predicting the cumulative income during the second lactation period. Differences in error metrics can be translated into economic terms using the observed variability of cumulative milk income (2225 CAD per cow per month), which was calculated based

on the actual cumulative milk income in the dataset. Therefore, the improvement achieved by our model corresponds to approximately 120–380 CAD per cow per month compared with Transformer-based models such as iTransformer and PatchTST. At the herd level (considering herd of 150 cows), this represents an estimated saving of 0.22–0.68 million CAD, annually. More accurate forecasts of cumulative milk income and their economic interpretation enable producers to assess expected cash flow ahead of time during the lactation period, which supports proactive adjustments in feeding strategies, calving schedules, and herd management planning. For instance, a predicted decline in income allows managers to reallocate feed resources, implement preventive health interventions, or change milking frequency to stabilize production and profitability. The associated economic improvements are reported in Tables 8.9 and 8.10.

Table 8.9 – Economic interpretation of predictive improvements for Lactation 2. MAE error values are translated into Canadian dollars (CAD) using $\sigma_{\text{income}} = 2225.38$. Annualized herd impact assumes 150 cows and 12 months.

Model	**MAE**	$\triangle$MAE (vs Bi-iGRU)	**Improvement (CAD/cow/month)**	**Annual Impact (CAD, 150 cows)**
Bi-iGRU	0.260	–	–	–
MuMu+Attention	0.257	-0.003	$ -6.68	$ -12,015
TimeMixer++	0.316	$+0.056$	$124.6	$224,280
SegRNN	0.315	$+0.055$	$122.4	$220,320
TimeMixer	0.291	$+0.031$	$69.0	$124,200
iTransformer	0.344	$+0.084$	$187.9	$338,220
S-Mamba	0.260	0.000	$0.0	$0
TimesNet	0.422	$+0.162$	$360.5	$649,000
DLinear	0.270	$+0.010$	$22.3	$40,140
PatchTST	0.430	$+0.170$	$378.3	$681,000

Bi-iGRU demonstrated competitive performance in forecasting sixth lactation milk income. Relative to SegRNN, iTransformer, and PatchTST, the observed MAE reduction (0.12-0.18) corresponds to 260-396 CAD per cow per month, representing an estimated saving of 0.46-0.71 million CAD annually for a herd of size 150 animals. These economic gains are derived from more accurate long-horizon forecasts of cumulative milk income, which allow improved resource allocation. Compared to the best competing model (MuMu+Attention), the difference in MAE was economically marginal (2.23 CAD per cow per month, indicating that Bi-iGRU maintains a competitive level of efficiency).

8.7 Conclusion

In this study, we developed a time series forecasting model tailored for dairy production and verified its effectiveness through comprehensive experiments. The results pinpoint its capability to efficiently capture and represent both temporal and cross-channel dependencies, achieving linear time complexity with respect to the number of time series channels. Leveraging a bidirectional GRU-based architecture, our model

Table 8.10 – Economic interpretation of predictive improvements for Lactation 6. Normalized MAE values are converted to Canadian dollars (CAD) using $\sigma_{\text{income}} = 2225.38$. Annualized herd impact assumes 150 cows and 12 months.

Model	**MAE**	$\triangle$MAE (vs Bi-iGRU)	**Improvement (CAD/cow/month)**	**Annual Impact (CAD, 150 cows)**
Bi-iGRU	0.315	–	–	–
MuMu+Attention	0.314	-0.001	$\$-2.23$	$\$-4{,}005$
TimeMixer++	0.316	$+0.001$	$\$2.23$	$\$4{,}005$
SegRNN	0.442	$+0.127$	$\$282.6$	$\$508{,}680$
TimeMixer	0.332	$+0.017$	$\$37.8$	$\$68{,}040$
iTransformer	0.431	$+0.116$	$\$258.2$	$\$465{,}000$
S-Mamba	0.318	$+0.003$	$\$6.68$	$\$12{,}024$
TimesNet	0.366	$+0.051$	$\$113.5$	$\$204{,}300$
DLinear	0.325	$+0.010$	$\$22.3$	$\$40{,}050$
PatchTST	0.493	$+0.178$	$\$396.1$	$\$712{,}980$

delivers accurate predictions for cumulative milk income, milk yield, and other dairy factors, empowering dairy farmers with valuable insights to optimize decision-making across lactation periods ahead of time. Furthermore, the model's design offers adaptability, accommodating diverse dairy datasets and potentially extending to other agricultural forecasting scenarios. By bridging deep learning with practical agricultural demands, this paper provides a baseline for future enhancements, such as refining predictions with multi-modal data, to further support sustainable and profitable dairy farming practices. Although Bi-iGRU demonstrates promising predictive performance on the Lactanet dataset, its generalizability to other regions, breeds, or management conditions remains to be validated and this represents one of the limitations of the current work. The model is designed to capture general temporal and inter-feature patterns, which suggests transferability to other datasets associated with different regions and even datasets from other domains. Future work will focus on external validation, and domain adaptation to ensure robust performance on more recent dairy datasets and analyzing broader geographic and production contexts.

CONCLUSION

Throughout this thesis we investigate the use of deep learning methods for multivariate time series forecasting in the context of dairy production, with a particular focus on predicting cumulative milk income and related dairy indicators. Recognizing the complex temporal and inter-variable dependencies inherent in dairy data, we formulated the estimation of dairy profitability as a multivarite time series forecasting problem. Our objective was to provide accurate and interpretable predictions that could assist in profitability estimation and strategic decision-making in precision livestock management. To support reproducibility and facilitate further research, the implementations of the models proposed in this thesis are publicly available at following github repositories :

- `https://github.com/bioinfoUQAM/MultiPatchFormer`
- `https://github.com/bioinfoUQAM/Bi-iGRU`
- `https://github.com/bioinfoUQAM/Forecasting_App`

To this end, we proposed several novel architectures specifically designed to address the unique characteristics of dairy data, as well as different public datasets from domains such as weather forecasting, traffic monitoring, and energy consumption. Among these, we introduced a multi-scale Transformer-based model capable of capturing temporal patterns at different granularities while representing cross-channel interactions. We also presented an inverted GRU model, designed to better capture dependencies along the channel dimension—a departure from the conventional recurrent modeling along temporal dimension. Building on this idea, we further extended the iGRU model into a bidirectional form (Bi-iGRU), enabling the capture of cross-channel dependencies, comprehensively. These architectures were rigorously evaluated on both public benchmark datasets and real-world dairy farm data.

Experimental results demonstrated that our proposed models consistently outperform, or remain competitive with, state-of-the-art forecasting approaches in terms of accuracy and computational efficiency. In particular, the Bi-iGRU model showed significant advantages in scenarios requiring nuanced cross-channel modeling, such as cumulative income forecasting and health indicator prediction, while requiring less memory and training time.

Beyond the methodological advancements, this thesis contributes to the growing body of work that supports data-driven herd management in the dairy industry. By integrating advanced deep learning models into the forecasting pipeline, we provide practical insights that can aid in optimizing resource allocation, improving animal replacement strategies, and ultimately enhancing farm profitability.

This research opens up several promising directions for future exploration and development. One of the prominent extentions involves adapting the proposed forecasting models for real-time prediction and continuous learning, enabling farms to dynamically update forecasts as new data becomes available and making the models responsive to rapid changes in herd health, feeding practices, or environmental conditions.

Another important direction is the application of federated learning across multiple farms or herds. In this context, models can be collaboratively trained without sharing sensitive or proprietary data, promoting data privacy while leveraging broader knowledge across diverse dairy operations. This approach could also enhance the generalizability of models across geographic regions, herd sizes, and management practices.

There are certain limitations associated with the proposed forecasting models. First, the presence of missing values in the input data, which are imputed using herd-wise or seasonal averages, might not fully capture the true underlying variability and could introduce biases that affect model performance. Second, there are some challenges introduced by the presence of noise and unpredictable variations in the data, including sudden health issues or environmental changes, which are difficult to anticipate and model precisely, affecting the model's forecasting accuracy. These factors can lead to deviations between the predicted and actual dairy production outcomes, particularly in more dynamic or less stable situations. Future work may also focus on integrating additional contextual and external variables, such as feed quality, weather conditions, energy usage, and market prices. These features can significantly enhance production and profitability but are often ingnored in current forecasting systems. Including such variables would allow for more comprehensive modeling that captures real-world economic patterns and operational complexities, leading to more practical and economically meaningful predictions. On the methodological side, future research could investigate hybrid modeling approaches, combining data-driven deep learning models with mechanistic or domain-driven models (e.g., biological or genetics based models) to capture both empirical patterns and theoretical knowledge. Finally, the forecasting pipeline could be generalized to support decision-making tools that recommend herd management actions, such as, culling, based on predicted outcomes, transforming forecasting from a passive analysis tool into an active decision making system for precision livestock farming.

Univariate and multivariate time-series methods to forecast dairy income

Vahid Naghashi, Gabriel M. Dallago, Abdoulaye Banire Diallo, Mounir Boukadoum

Departement d'Informatique, UQAM, Montreal, QC, Canada
PO Box 8888 Downtown station, Montreal, QC H3C 3P8, Montreal, Canada
diallo.abdoulaye@uqam.ca

Abstract

Forecasting the income from milk sales can be addressed as a time-series problem since the sequence of multiple dairy attributes during lactation cycles are inter-related and temporally dependent. In this paper, we provide a framework to forecast the income from milk sales during the third lactation of the dairy cows based on dairy attributes recorded through the first and second lactation. We modeled the problem as univariate and multivariate time-series predictions. We propose several state-of-the-art implementations with ARIMA, N-BEATS, transformer and an original method, MuMu+attention, that combines Long-Short Term Memory neural network and attention mechanism to capture the temporal dependencies. To benchmark the implemented methods, we curated data from 147,749 dairy cows from 5,844 Canadian herds. The monthly income from milk sales ($CAD) measured at each cow during their third lactation was treated as the prediction target. The dataset was composed of dairy attributes of milk quality, production, season, year, and health recorded over the first and second lactation of the dairy cows. The results highlighted that most of the methods can achieve relative good performance with the best prediction accuracy obtained by MuMu+attention. MuMu+attention results were 43% better over the classic ARIMA model. By forecasting the income from milk sales, our model could help farmers to early identify less profitable animals and better allocate resources.

Introduction

Dairy farming is one of the largest sectors in agriculture that has been under study through data-driven and machine learning methodologies. Promising results were obtained in automatic cropping of the cow's body region and cow's pattern identification for individual animals (Zin et al. 2018). In another work, deep convolutional neural networks were used for detection of the key parts of a dairy cow's body, resulting in an accurate detector (Jiang et al. 2019). Promising results were obtained by exploiting a deep learning model for calving prediction from activity, lying, and ruminating behaviors of dairy cattle (Borchers et al. 2017). Therefore, the use of machine learning and deep learning based models lead to promising results and its application to predict dairy production could yield satisfying results (Frasco et al. 2020).

Income from milk sale is the main factor associated with the profitability of a dairy farm. A forecasting tool would allow farmers to optimize the allocation of resources by early identifying and removing less profitable animals from the herd. The problem of forecasting income from milk sales can be modeled as a univariate time-series task since the lactation cycles are inter-related and temporally dependent, or multivariate since milk production, the consequently income, is affected by heath, productivity, environmental, and management conditions. Among the classical time-series prediction methods, Autoregressive Integrated Moving Average (ARIMA) has shown a good performance in univariate time-series prediction tasks (Contreras et al. 2003). Recently, deep learning models have been exploited in time-series prediction domain. N-BEATS (Oreshkin et al. 2019) is one of the well-known deep prediction models, in which residual connections are used for univariate time-series prediction and the model's architecture is based on a very deep stack of fully-connected layers. In MuMu (Frasco et al. 2020), Long-Short Term Memory (LSTM) network was exploited in the dairy forecasting field and gave rise to auspicious results.Recently, Transformer models (Vaswani et al. 2017) got more attention of researchers due to their ability to represent the long-term temporal dependencies efficiently by incorporating multi-head self attention in their structure. The efficacy of the self-attention layers gave us the hint about using an attention module in our prediction framework.

The objective of this paper is twofold: first, we propose an extension of MUMU using LSTM and an attention mechanism to forecast the income from milk sales; second, we benchmark univariate models against multivariate ones in predicting future profit using a well curated data from 147,749 dairy cows.

Preliminary

The problem of lifetime milk revenue prediction can be posed as follows. For each input example (ith cow sample) of length T, i.e. $x_i = (x_i^1, ..., x_i^T) \epsilon \Re^{p \times T}$ with p as the number of input dairy factors and T the length of the time-series (total length of the first and second lactation fixed for all the cow samples), a prediction model forecasts the upcoming milk revenues of M steps (months) ahead in third lactation, $\hat{r_i} \epsilon \Re^M$. Therefore, the goal is to learn a function f which maps the input multivariate time-series $X \epsilon \Re^{s \times T \times p}$ to the estimated milk income values in the future lactation months

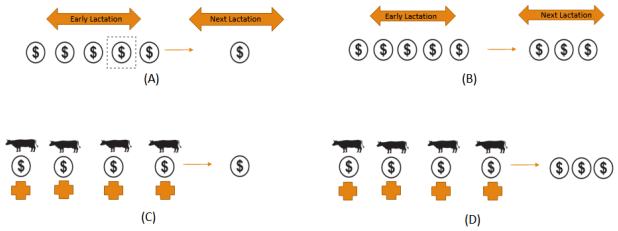


Figure 1: Univariate and multivariate statement of our dairy prediction problem.

$\hat{r} \epsilon \Re^{s \times M}$, with s being the number of the cow samples presented to the model as the training set: $\hat{r} = f(X)$.

Methodology

Prediction of dairy income can be stated as four different cases (Figure 1): univariate or multivariate inputs, single or multiple outputs. In the first case, the input is a sequence of input dairy incomes during the early lactation cycles and the output is the value of the next income in the next lactation. The second case is similar to the first one, with the difference that the output is multiple dairy incomes. In the third case, the input window consists of multivariate series associated with the multiple dairy factors through the early lactation and the output window corresponds to a single income in the next lactation. Finally, the input window in the fourth case contains multivariate series of multiple dairy attributes similar to the latter case, and have a sequence of dairy incomes as its outputs.

Model Architecture

Here we propose MuMu+Attention, which builds on top of the MuMu model (Frasco et al. 2020). It implements time-series of the individual dairy attributes corresponding to earlier lactations. It consists of two LSTM layers stacked on top of each other and one attention layer, followed by a linear layer acting as a decoder. The architecture of the proposed framework is illustrated in Figure A1.

For the purpose of using all the hidden states and temporal feature selection, an attention layer (Vaswani et al. 2017) has been embedded on top of the last LSTM layer in our model. The attention layer consisted of two linear layers and one $tanh$ activation function in between. The final attention weights were the result of applying a $softmax$ function to the output of the second linear layer to normalize the attention scores. These weights were multiplied by the corresponding hidden states and a weighted hidden vector was calculated in order to be fed into the final linear layer which generates the final prediction over the target window.

The corresponding formulas describe the components of the attention layer:

$$L_1 = tanh(W_1 x + b_1) \quad (1)$$

$$Attention\_Outputs = Softmax(W_2 L_1 + b_2) \quad (2)$$

In the above formulas, x represents the output of the last LSTM layer with the shape of (batch_size, sequence_length, hidden_size). $W1$, $W2$, $b1$ and $b2$ are learnable parameters which have been trained in an end to end manner together with the other parameters of the model.

The major advantage of using the attention layer is that the information in all of the hidden states has been exploited, rather than using only the output of the last LSTM cell and this can be referred to as a temporal feature selection step which learns and assigns the importance weights to the sequence of the hidden states associated with the input window (input time steps). A dropout layer is used in the output of the second LSTM layer in order to avoid over-fitting. The value of the dropout rate hyper-parameter determines the ratio of the hidden states whose outputs are dropped out and this hyper-parameter is set to 0.5 in our model. Finally, the decoder in our model, which is a linear layer with no activation function, generates the output predictions by one forward pass instead of the time consuming dynamic decoding used in the conventional architectures. In our work, the mean squared error was used as the loss function while training the model.

Experimental settings

Data

The input to our prediction model was a multivariate time-series containing a set of dairy attributes, including metrics of milk quality, seasonality, year, health, and management factors, recorded during the first and second lactation of 147,749 dairy cows from 5,844 Canadian dairy farms over the years of 2006 to 2017. The prediction targets were the monthly income from milk sales ($CAD) measured at each cow during their third lactation.

Data Pre-processing

In the experiments, lactation length was fixed to 11 months for first, second, and third lactation based on the mean + one standard deviation of the lactation lengths related to the training cow samples. For cows with lactation lengths shorter than 11 months, additional data rows were created in the missing months and imputed through linear interpolation based on the two closest months. The following steps were taken to clean the data: 1- Keeping only animals having test records in the first, second and third lactations, 2- Deleting the records from the dry period: dry period is defined as the months in which a cow doesn't produce milk (which mostly occurs between two lactation cycles) and milk value (income) is almost zero. Since there is no income from sales during the dry period, records associated with these months were deleted, 3- Removing the animals who left their herd before and during the third lactation, 4- Deleting duplicate records, 5- Deleting the records with negative milk value and cumulative milk value: The reason for removing such records, which constituted a small percentage of the data, was to remove the inconsistencies in the data set, as the milk value is the income from milk sale (CAD) and negative values of dairy income could be accounted for as an inconsistency due to the errors in data acquisition steps., 6- Deleting the records which included contradictions, e.g., rows indicating the cow was milking, while the milk yield of those

records was equal to zero, 7- Outlier removal (their corresponding rows): The values outside the range of Mean -/+ 2.5 $\times$ standard deviation were specified as outliers for a given dairy attribute and were deleted. All the dairy attributes used as input to our model and their descriptions are shown in Tables A1 and A2 (Appendix).

Among the 147,749 dairy cows, 100,000 were selected to train and the remaining 47,749 cow samples to test the models. Missing data was imputed after train-test split to avoid information leakage (Thomas et al. 2020). In our work, missing value imputation were performed in 7 consecutive steps: first the cow samples were grouped based on their herd ID, season and year, then the missing values within each group were imputed using the average of non-missing samples within the same group. The imputation process in the remaining six steps was similar as in the first one, with the following attributes used for grouping, respectively: (herd ID, season), (herd id, year), (herd id), (season, year), (season), (herd id).

Experimental Details

Baselines We choose five forecasting methods as comparison models: two univariate (ARIMA (Contreras et al. 2003) and N-BEATS (Oreshkin et al. 2019)) and two multivariate models (MuMu (Frasco et al. 2020) and a standard Transformer model (Vaswani et al. 2017) adapted for time-series prediction. Most of the above methods are well known in the time-series domain. The ARIMA model is chosen as a baseline method as it is a classic statistical approach in time-series analysis. The parameters used in the ARIMA are taken from (Frasco et al. 2020) which used a stepwise algorithm to determine p, q, P and Q parameters. The $N-BEATS$ model (Oreshkin et al. 2019) had four blocks. The first two blocks had a trend basis function in their outputs, and the other two contained a seasonality basis. The number of fully-connected layers inside each block was fixed to 4 with hidden-size of 128. As the N-BEATS model is designed to receive a univariate sequence, we fed the sequence of milk incomes in 22 months as its input. The parameters for the MuMu model are: 2 LSTM layers, one linear layer (without an activation function) and the hidden size is fixed to 32 in all layers. The Transformer model which was used as another baseline method in our experiments, was composed of two encoders and one decoder layer, in which the input of the decoder was the last time step of the input window.

Grid Search for Hyper-parameters determination Grid search was conducted to determine the hyper parameters (batch size, learning rate and hidden size of the LSTMs) due to its common usage in other related works and satisfying results (Zhou et al. 2021). Our proposed model was optimized with Adam optimizer and learning rate of $1e^{-4}$. The total number of epochs was set to 20 with an early stopping based on the validation loss. **Setup:** All the numerical inputs were standardized with zero mean and unit standard deviation. At the same time, categorical variables were normalized to the range between 0 and 1. The above transformation was applied to each dairy feature, separately.

RMSE results				
Univariate		Multivariate		
ARIMA	N-BEATS	Transformer	MuMu	MuMu+Attention
7.094	4.198	4.064	4.082	**4.052**

Table 1: Prediction results in terms of RMSE (teste dataset).

MAE results				
Univariate		Multivariate		
ARIMA	N-BEATS	Transformer	MuMu	MuMu+Attention
5.560	3.256	3.178	3.175	**3.143**

Table 2: Prediction results in terms of MAE (test dataset).

The input window was set to 22 (with the length of 11 for both of the first and second lactation). The prediction (target) window size (length of the third lactation) was fixed as 11 in our experiments according to the mean of the lactation lengths corresponding to all the cow samples. We evaluated our prediction framework using the $RMSE = \sqrt{\frac{1}{N_{cows} \times N_{months}} \sum_{m \epsilon months} \sum_{c \epsilon cows} (\hat{p}_{c,m} - p_{c,m})^2}$ and the $MAE = \frac{1}{N_{cows} \times N_{months}} \sum_{m \epsilon month} \sum_{c \epsilon cows} |\hat{p}_{c,m} - p_{c,m}|$ on the forecasting window. All the models were trained and tested on 4 NVIDIA T4 Turing GPUs with 16 GB GDDR6 memory. Additionally, multiple pairwise Wilcoxon tests, with Bonferroni p-value adjustment, were used to compare models.

Results and Discussion

Tables 1 and 2, summarize the forecasting accuracy of all the methods on the test dataset. The best results are highlighted in boldface in each setting. We also reported the chosen hyper-parameters (the best combination), including hidden size, learning rate and batch size after performing the grid search in Table A3 (Appendix).

MuMu+Attention model was able to forecast the income with the highest accuracy based on the RMSE (Table 1) and MAE (Table 2) measures. The LSTM layers included were able to represent and capture the long-term temporal dependencies (Gers, Schmidhuber, and Cummins 2000) by exploiting the input, forget, update, and output gates in its structure. Those gates help the LSTM to select the most relevant information and update the previous state using the current input at each time step. At the same time, forget and update elements give the LSTM the capability of remembering the long-range dependencies and making use of such relationships in the prediction process.

Among the baseline methods, the classical ARIMA forecasts had the highest error. The linear nature of this model hinders its ability to capture more complex and non-linear temporal correlations. The N-BEATS model had a relatively higher error compared to the multivariate approaches, which is likely because it does not have a sequential module in its structure to represent the temporal dependencies. Capturing long-range dependencies is a key factor in time-series prediction (Zhou et al. 2021) and all the LSTM or Transformer based approaches try to represent such relations by captur-

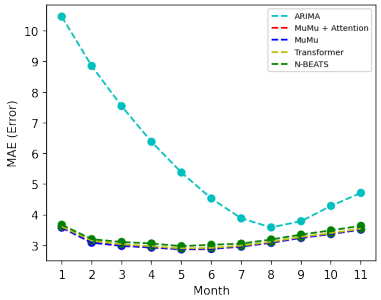


Figure 2: Monthly Errors (MAE) of different methods and our proposed framework on prediction of the milk incomes.

ing the impact of different time steps on each other, which seems to be a dominant factor for the success of those methods. The reason behind the relatively lower performance of the Transformer model might be due to the lack of sufficient and enough data for training of this model, which has more parameters than the other models (Table A7). As the errors related to different deep learning models are in the same range, the training and inference time of the MuMu+Atten and other models are reported in table A4 to give an insight on the time complexity of the deep learning based methods. Based on the results, the N-BEATS model needs more training and inference time than the MuMu+Atten, despite its lower performance compared to the multivariate models. We further conducted other experiments to predict the milk value in a single month after the first and second lactation (input window). The results indicate that the model nearly gave the same results as the prediction of the multiple months (Table A5).

Figure 2 depicts the forecast accuracy of the proposed framework in comparison to the other models over each month of the third lactation. Except for the ARIMA model, the best forecasts occurred in the middle of lactation (month 5) compared to the beginning and the end. This is likely because there is a great variability among cows in the amount of milk produced and consequently sold during these steps, making it more difficult to forecast. The distribution of MAE for all models at the herd level is plotted in Figure 3. There was no strong evidence of herd-bias as the distributions were visually similar. The MAE, at the herd level, was not statistically different between the multivariate models ($p \geq 0.05$) and it was lower than both univariate models ($p < 0.05$; Table A6). This indicated that the proposed framework could also capture the long-range temporal dependencies in input and output windows by processing the input sequence using the LSTM layers and representing the importance of each time step through the attention weights. Furthermore, the generative style decoder (the output linear layer in our framework) acquired the output predictions in one forward pass and avoided the error accumulation during the testing phase.

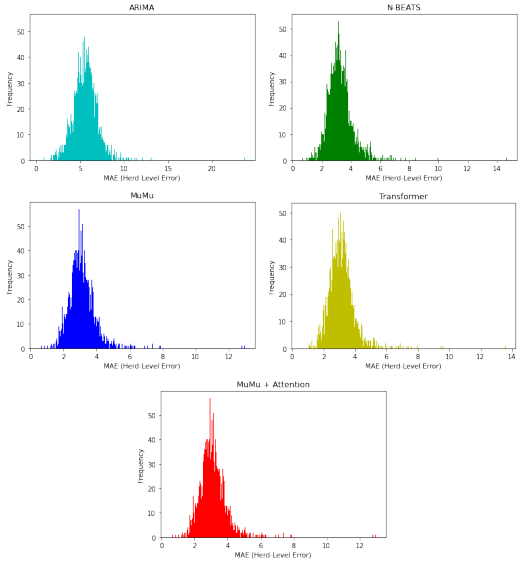


Figure 3: Herd-based Error (MAE) distribution of different methods.

Conclusion

In conclusion, we studied the problem of forecasting the income from milk sales by designing a framework and comparing univariate and multivariate approaches. We enriched our framework with the MuMu+Attention model that combines LSTM and attention mechanism. The experimental results showed that multivariate models tend to perform better even though the performance of NBEAST can be an important way of approximating the profit just using a univariate time-series. However, we showed that MuMu+Attention provided the highest accuracy.

References

Borchers, M.; Chang, Y.; Proudfoot, K.; Wadsworth, B.; Stone, A.; and Bewley, J. 2017. Machine-learning-based calving prediction from activity, lying, and ruminating behaviors in dairy cattle. *Journal of dairy science*, 100(7): 5664–5674.

Contreras, J.; Espinola, R.; Nogales, F. J.; and Conejo, A. J. 2003. ARIMA models to predict next-day electricity prices. *IEEE transactions on power systems*, 18(3): 1014–1020.

Frasco, C. G.; Radmacher, M.; Lacroix, R.; Cue, R.; Valtchev, P.; Robert, C.; Boukadoum, M.; Sirard, M.-A.; and Diallo, A. B. 2020. Towards an Effective Decision-making System based on Cow Profitability using Deep Learning. In *ICAART (2)*, 949–958.

Gers, F. A.; Schmidhuber, J.; and Cummins, F. 2000. Learning to forget: Continual prediction with LSTM. *Neural computation*, 12(10): 2451–2471.

Jiang, B.; Wu, Q.; Yin, X.; Wu, D.; Song, H.; and He, D. 2019. FLYOLOv3 deep learning for key parts of dairy cow body detection. *Computers and Electronics in Agriculture*, 166: 104982.

Oreshkin, B. N.; Carpov, D.; Chapados, N.; and Bengio, Y. 2019. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*.

Thomas, R. M.; Bruin, W.; Zhutovsky, P.; and van Wingen, G. 2020. Dealing with missing data, small sample sizes, and heterogeneity in machine learning studies of brain disorders. In *Machine learning*, 249–266. Elsevier.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 11106–11115.

Zin, T. T.; Phyo, C. N.; Tin, P.; Hama, H.; and Kobayashi, I. 2018. Image technology based cow identification system using deep learning. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, 236–247.

Appendix

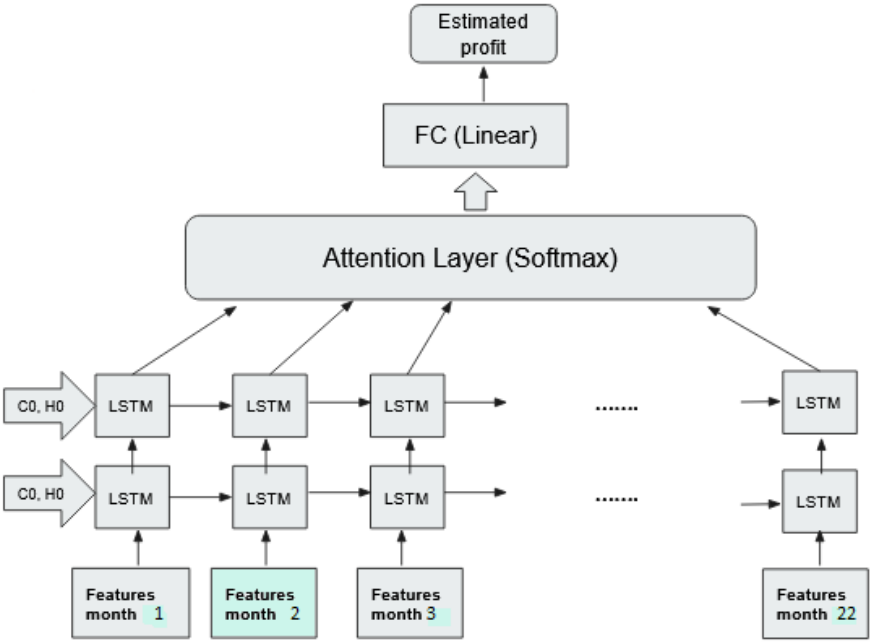


Figure A1: Architecture of the proposed prediction framework.

Variable	Missing %	Min	Max	Mean	Std
Milk produced in 24 hours (Kg)	1.4	0.20	61.40	29.60	9.25
Milk income (CAD)	0.002	0.09	42.95	21.21	5.94
Somatic Cell Count (1000/mL)	17.62	1	1754	156.59	254.87
Milk Urea Nitrogen	56.04	2.10	19.80	10.91	3.27
Lactose yield (Kg)	37.63	0	6.30	4.55	0.22
Fat yield in 24 hours (Kg)	1.64	0	2.40	1.19	0.38
Protein yield in 24 hours (Kg)	1.64	0	1.94	1.00	0.30
Number of days cow has been in milking	0	0	517	169.56	99.39
Fat to protein ratio	1.64	0.73	1.67	1.20	0.16

Table A1: Numerical dairy variables used in our framework.

Categorical variables				
Variable	**Missing %**	**Levels**	**Number of observations**	**% of each level**
Lactation Number	0	1	1625239	33.33
		2	1625239	33.33
		3	1625239	33.33
Number of milking per day	0	1	163201	3.33
		2	4650067	95.37
		3	62449	1.28
Test season	0	1	1235340	25.33
		2	1226102	25.14
		3	1220232	25.03
		4	1194043	24.49
Birth season	0	1	1109460	22.75
		2	1325577	27.18
		3	1274922	26.14
		4	1165758	23.91
Animal condition	0	2	4846703	99.40
		4	29014	0.06
Test year	0	6	155074	3.18
		7	304329	6.24
		8	443724	9.10
		9	486607	9.98
		10	472280	9.68
		11	479705	9.83
		12	498897	10.23
		13	511067	10.48
		14	521706	10.70
		15	486234	9.97
		16	331883	6.81
		17	170715	3.50

Table A2: Categorical dairy variables used in our framework.

Hyper-parameter	Selected value
Hidden size	32
Learning rate	$1e^{-4}$
Batch size	1
Epochs	20

Table A3: The selected hyper-parameters in training of our proposed model

ARIMA	N-BEATS	Transformer	MuMu	MuMu+Atten
$>$ 12 hours	4 hours, 7 mins	2 hours, 34 mins	1 hour, 36 mins	2 hours
$O(L * p^3)$	$O(c * L * d)$	$O(L^2 * d)$	$O(L * d^2)$	$O(L * d^2)$

Table A4: Running time of different models (Training + Inference time). In this table, L is the length of the time series, d in the model's dimensionality, c is a multiplier, and p is the order of the ARIMA model.

	One month RMSE	One month MAE
N-BEATS	4.72	3.72
Transformer	4.58	3.60
MuMu	4.57	3.62
MuMu+Attention	4.56	3.61
ARIMA	13.53	11.81

Table A5: RMSE and MAE of the forecasting the milk income (value) in a single month ahead of the input window (first and second lactation)

	ARIMA	N-BEATS	Transformer	MuMu
N-BEATS	< 0.001			
Transformer	< 0.001	< 0.001		
MuMu	< 0.001	< 0.001	0.05	
MuMu+Attention	< 0.001	< 0.001	0.05	1.00

Table A6: Pairwise comparisons of the mean absolute errors for all models at the herd level using the Wilcoxon rank sum test. The Bonferroni method was used to adjust the p-values for multiple comparisons.

Model Name	Number of parameters
N-BEATS	109325
Transformer	276587
MuMu	15339
MuMu+Attention	16428
ARIMA	3

Table A7: Number of parameters of different models

Journal Title Here, 2022, pp. 1–9

doi: DOI HERE
Advance Access Publication Date: Day Month Year
Paper

PAPER

Transformer-BEATS: A Transformer model for time series prediction of dairy milk production

FOR PUBLISHER ONLY Received on Date Month Year; revised on Date Month Year; accepted on Date Month Year

Abstract

Predicting the milk production of dairy cows is important for precision livestock management. It can be achieved by observing the historical volume of each cow and features encoding its health status and the farm environment. This problem can be modeled as a multivariate time-series forecasting task, which demands capturing temporal dependencies and inter-series relations effectively. The current time-series models for dairy forecasting often fall short to capture these intricate temporal dependencies and inter-series (spatial) relationships. Here, we present a Transformer-based architecture that effectively captures these spatial and temporal dependencies through self-attention components, with a new adaptive way to integrate the forecasts from the temporal and spatial encoders. The proposed model is enhanced by connecting two Transformer encoder blocks in a back-cast (residual) manner, ensuring that the most relevant information is provided to the following block. Our approach yields better results, particularly in terms of average prediction error per cow, than the state-of-the-art methods when predicting milk production in upcoming lactation of 47,000 cows.

Key words: Milk production, Transformer, Inter-series correlation, Temporal dependencies

Introduction

Over the past decades, global food consumption has seen a significant surge, creating a pressing need for farmers to enhance their production efficiency [2]. Among the key agricultural sectors, dairy farming stands out with its substantial contribution to milk production. Recognizing the vital role of the dairy industry in both human life and agriculture, it is of paramount importance to equip farmers with the latest technological tools to help achieve their goals [4]. These tools also reduce costs and increase the overall production efficiency, not only for the farmers' benefit, but also to ensure a reliable and sustainable supply of dairy products to meet the growing demand by the world population. Deep learning methods provide reliable predictions about calving, disease and productivity, and farmers can make informed decisions about their livestock management, such as identifying low-performing cows and taking strategic steps to optimize their herd sizes [1] [15].

The problem of dairy production (income) estimation can be stated as a multivariate time series prediction task, based on multiple dairy features recorded periodically over the lactation months (on test days). The dairy features are categorized into health, management, milk quality, milk value, farm environmental and seasonal factors.

The general field of time-series prediction has captured the interest of many researchers in recent years, and numerous deep learning-based models have been developed. In this paper, a novel model for dairy series prediction is described. The model is designed to predict the milk income in future lactation by taking into account the historical information, in continuation of previous related works [10] [4] about dairy profitability prediction. The forecast generated by our model can offer valuable assistance for dairy farmers to make informed decisions about their livestock, such as whether to retain or cull cows from their herd. Our model is based on the Transformer model [16], with architectural modifications to account for the temporal relations within the sequence of lactation months and the dependencies between the sequences of dairy variables recorded on specific days during the lactation months. Overall, the following contributions to the state-of-the-art (SOTA) have been made:

- The first Transformer-based model to predict dairy production;
- A novel Transformer design that features:
  - The efficient capture of intra-series (temporal) and inter-series (spatial) correlations, with the model's prediction based on the weighted sum of both results;
  - The use of backcast encoder connections to help each processing block focus on the essential portion of its input sequence.

Predicting the milk income during the upcoming lactation months involves tackling a multivariate time series prediction task, leveraging historical lactation data and pertinent dairy features. This type of forecasting requires the analysis of sequential data to discern trends and patterns, facilitating the learning process to predict future values of one or more variables of interest.

Related Work

In more recent years, the dairy industry has witnessed a proliferation of deep learning-based approaches. For instance, the Long-Short Term Memory (LSTM) neural network has been exploited for cow's milk profitability prediction [4]. It relied on an integrated multiple source test-date information to predict the profits in future months. A deep learning framework is proposed in [7] to encode the dairy features, then predict the latent representation of milk yield sequence and generate the lactation curve by leveraging Convolutional Neural Networks (CNN) and encoder-decoder modules. The prediction of first test day milk yield is accomplished through the application of classical machine learning models, including random forest in [13]. This is achieved by giving the models test day production, reproduction and herd features. Models for time series forecasting can be generally categorized into conventional statistical models and deep learning models. Statistical models, such as ARIMA (Auto-regressive Integrated Moving Average) has shown a good performance [3]. ARIMA corresponds to a class of linear models and its main shortcoming is that it is not applicable to non-linear systems and is a univariate model, not capable of utilizing other covariates in the prediction process. [21]

One of the well-known deep learning models called N-BEATS was proposed for univariate time-series forecasting [12]. It is based on backward and forward residual connections and a deep stack of fully-connected layers which is fast to train and is interpretable thanks to the trend and seasonality basis components. In order to capture short and long term temporal information in time series, LSTNet was proposed [5], which exploits Convolutional Neural Network and Recurrent Neural Network to capture short term local dependencies between the variables and long term patterns of time-series trend.

The Transformer architecture has shown promising performance in sequence processing and time-series prediction, and many time-series forecasting models have been proposed that utilize with some modifications. For instance, the work in [19] designed a deep Transformer model that includes both the encoder and decoder modules to forecast Influenza prevalence. The Informer, a variant of Transformer model adapted for time-series forecasting introduced in [23], exploits a specific attention mechanism called "probSparse self-attention" along with a distillation mechanism. This hybrid approach emphasizes dominant attention scores and reduces computational overhead and memory consumption, making Informer well-suited for efficient processing of long time-series data. The Autoformer [18] and FEDFormer [24] utilize the Fourier transform to extract frequency information and combine it with time domain features inside the Transformer architecture. Recently patch-wise encoding has been most popular and proven to be useful in capturing temporal relations in long time-series by using Transformer architecture. The Crossformer model [22] captures cross-time and cross-dimension dependency by utilizing Dimension-Segment-Wise embedding and a two-stage attention mechanism to capture the correlations between the input series and the dependencies along the time dimension. Another well-known model based on patch-wise attention is the PatchTST [11] which uses channel independence and patch-level attention for data mining, however it doesn't capture the relations between different variables. More recently, [9] proposed iTransformer which utilizes embedding along time dimension and then extracts the dependencies between variables through self-attention and feed-forward modules.

Nevertheless, both temporal and spatial (inter-series) dependencies can be extracted independently and then integrated into the final prediction. Moreover, connecting the layers in a residual fashion helps the next layer to focus on the more relevant signal portion. In this work, we have taken into account these aspects in order to enhance the performance of the Transformer model for prediction of dairy time-series.

Transformer and Multi-head Self-Attention Module

The Transformer model has proven its efficiency in capturing long-term dependencies in a sequence (sentence or time-series), thanks to its self-attention block which provides the correlations between different positions in a sequence. This allows to determine the importance of each input word or token to generate the output representation, and focus on the relevant elements. Moreover, a multi-head attention mechanism allows to process the input sequence from several viewpoints, enabling the model to capture different types of relationships (in different spaces) in the input sequence. To compute the multi-head self-attention, the input sequence stack is first converted into so called queries, keys, and value matrices through linear projections:

$$Q_i = X \cdot W_i^Q, \quad K_i = X \cdot W_i^K, \quad V_i = X \cdot W_i^V \tag{1}$$

In the above formulas, X represent the embedded input sequence, Q_i, K_i, and V_i are the projection results, i.e., the query, key, and value matrices, and W_i^Q, W_i^K, and W_i^V are the corresponding projection matrices for ith head. These matrices essentially map the input tokens into a common space where they can be compared. Each projection matrix is of size $d_{model} * d_k$, where d_{model} is the input dimension (embedding dimension) and d_k is the head dimension. The input multivariate time-series is represented in a tensor including the samples with the variables (series values) over the input time steps. Thus, X is of dimension $(Batch, Length, d_{model})$. In this context, "batch" is referred as a collection of multiple samples, "Lengths" corresponds to the length of input time-series, and d_{model} signifies the dimension of the feature space in which the attributes are projected.

The head dimension is calculated as $d_k = d_{model}/h$, where h is the number of heads. The projected queries and keys are multiplied, and the result is scaled down by d_k before feeding it to a softmax function in order to obtain the attention scores:

$$Attention(Q, K, V) = softmax\left(\frac{Q \cdot K^T}{\sqrt{d_k}}\right) \cdot V \tag{2}$$

Dividing the dot products of queries and keys by the square root of the head dimension ensures stability during training and controlling the scale of the attention scores.

As mentioned, the input undergoes h projections into head space, each time with an independent self-attention computation:

$$head_i = Attention(XW_i^Q, XW_i^K, XW_i^V) \tag{3}$$

Then, the results are concatenated to represent the temporal dependencies as seen by the different attention heads, hence providing valuable information to the downstream blocks or tasks. Finally, the concatenated attention scores are linearly projected by a W^O matrix of size $d_k * d_{model}$ to convert them into an output that can be processed by additional transformer layers:

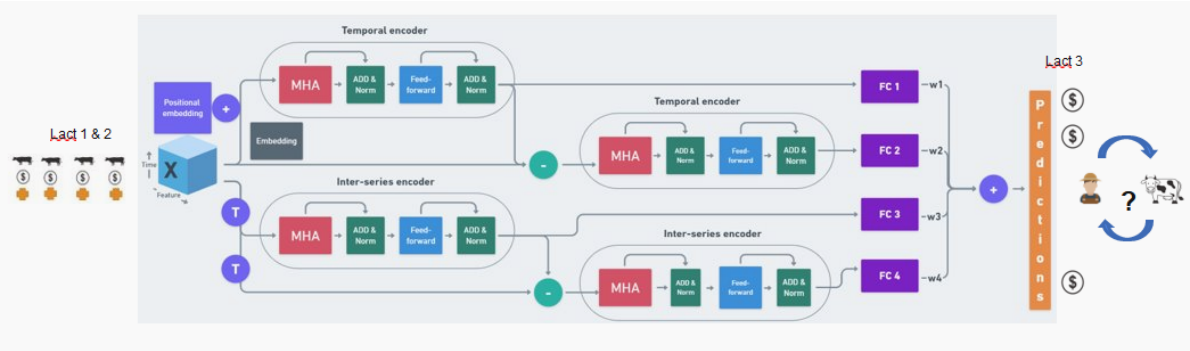


Fig. 1. The proposed Transformer based model with inter-series (spatial) self-attention modules. The input of the temporal encoder part is embedded and positional encoded in order to inject the sequence order information to the input series. The second temporal and spatial encoders take the residual inputs, which are obtained by subtracting the output of the previous block from its input. This helps the next encoder block to focus on the relevant part of the time-series signal, which is achieved by subtracting the previously encoded information from the input signal, passing only the residual part to the next block for further utilization in the prediction process. The final prediction is the result of a weighted sum of the predictions obtained from each encoder after flattening and passing through the separate linear layers.

$$MultiHead(Q, K, V) = Concat\,(head_1, \ldots, head_h) \cdot W^O \quad (4)$$

Methods

Problem Definition

The problem of milk production prediction can be stated as follows: given a herd of cows of size N, each cow i represented by a set of records $\{\mathbf{x}_j^{(i)} = (x_{j,1}^{(i)}, ..., x_{j,T}^{(i)})\}_{j=1..v} \epsilon\, \Re^{v \times T}$, where $\mathbf{x}_j$ is a sequence of dairy variables, v represents the number of dairy variables considered and T is the record length (total length of the first and second lactation periods, fixed for all of the animals), predict M steps (months) of the upcoming milk production in the third lactation: $\hat{r}^{(i)} \epsilon\, \Re^M$. Thus, the goal is to learn a non-linear function f which, given the input multivariate time-series $X \epsilon\, \Re^{N \times T \times v}$, estimates milk income values in the future lactation months $\hat{r} \epsilon\, \Re^{N \times M}$, with the N cow samples presented to the model as the training set.

Inter-series Correlations

The multiple series related to the different source variables impact each other and are correlated [22]. Therefore, capturing those dependencies is important for better forecasting. To achieve accurate results, it is essential to explicitly model the intricate relationships between the variables. The existing time-series forecasting methods often fall short in capturing these complex inter-dependencies, leading to sub-optimal predictions. However, the self-attention mechanism can also capture inter-series dependencies, considering the temporal information [8]. In this case, instead of calculating the attention scores at each time-step, the correlations between the series can be captured by calculating the attention importance scores for each pair of variables. For this purpose, the input multivariate time-series is transposed before feeding it to the multi-head self attention module. The similar steps, including projection, attention score calculation is performed using the time dimension interchanged with the variable (series) dimension in order to obtain the importance scores for each pair of the variables. Therefore, the input to the inter-series (spatial) self-attention module is the transposed input sequence with the shape of $(Batch, Feature, Length)$ without adding any positional information.

It is important to mention that the inter-series self-attention module receives the transposed input multivariate series without embedding. This approach is chosen deliberately to maintain low time complexity and preserve the original number of input time-steps, because the embedding changes the number of time-steps when applied on the time dimension. In the projection formulas below, X is the transposed input multivariate series without embedding. W_{int}^Q, W_{int}^K and W_{int}^V are of dimensions $d_T * d_T$ with d_T indicating the time dimension size (number of the input time steps).

$$Q_{int} = X \cdot W_{int}^Q, \quad K_{int} = X \cdot W_{int}^K, \quad V_{int} = X \cdot W_{int}^V \quad (5)$$

The inter-series attention scores are calculated in the same way as in the standard self-attention:

$$Score_{int} = softmax\left(\frac{Q_{int} \cdot K_{int}^T}{\sqrt{d_T}}\right) \quad (6)$$

$$Attention(Q_{int}, K_{int}, V_{int}) = Score_{int} \cdot V_{int} \quad (7)$$

In order to keep the computational complexity low, multi-head attention is reduced to single head self-attention in the inter-series attention module.

Residual Blocks

The residual blocks within the N-BEATS [12] model plays a crucial role in enhancing prediction accuracy across successive blocks. These blocks facilitate the selective transfer of relevant information, allowing only essential signals to be forwarded to the subsequent blocks. As a result, each subsequent block can focus and build upon the remaining pertinent details passed down from the previous block, contributing to an overall improvement in the model's forecasting capabilities. Inspired by the forecast and back-cast outputs design, the successive encoder blocks in our model are connected using the back-cast inter-connection, which is the residual part of the latest encoder output obtained by subtracting the output of the previous encoder module from its input.

Proposed Model

The architecture of the proposed Transformer based model is illustrated in Figure 1. This model exploits the power of the

self-attention mechanism to extract meaningful temporal and spatial (inter-series) correlations. Two separate Transformer encoder blocks are the main building components. The first encoder block consists of the multi-head self-attention and feed forward components connected to each other along with residual connections. Each component is followed by a normalization layer. This prevents covariance shift across layers during training, which may cause gradient problems and their negative effect on convergence. The second parallel encoder block has the same structure, except the input sequence, where the original input time-series is transposed to swap the time and feature dimensions. Then, the ensuing attention scores indicate the pairwise inter-variable dependencies at each time step. In this paper, the network is of depth 2, since two successive parallel temporal and series-wise encoder blocks constitute the model. More layers can be added depending on data availability and dataset attributes.

In the proposed model, all the encoder outputs contribute to the final prediction. The encoders in the first layer extract shallow features (both temporal and inter-series ones), while the encoders in the second layer capture more complex patterns. Therefore, combining all those outputs contribute to obtain better results. The encoder outputs are flattened and fed into separate linear layers (fully-connected with no activation function) for producing separate predictions, with different outputs contributing to a weighted sum that gives the final prediction:

$$\text{Prediction} = \sum_{i=1}^{4} w_i \cdot o_i \tag{8}$$

In which o_i is the prediction from the ith encoder and w_i is the associated weight. The use of a weighted sum for the final prediction allows to account for the relative importance of the different outputs that contribute to it, which should lead to better forecasts and facilitates the interpretation of the final prediction. The weights are learned like any other network parameters in an end-to-end manner. They are randomly initialized and then normalized using the softmax function. We called our model Transformer-BEATS as it exploits the power of the Transformer [16] blocks and the back-cast or residual connections introduced in the N-BEATS model [12] in an adaptive architecture.

Results

The selected dairy dataset for training consists of the records of 147,749 dairy cows from 5,844 Canadian dairy farms, recorded on test days during the first and second lactation periods from 2006 to 2017 [10]. Each record includes metrics of milk quality, seasonality, year, health, and management factors. The dataset was curated from 1.48 million animals after animal selection, outlier removal and missing data imputation. More information on the dataset can be found in the appendices section.

Experimental Settings

The dataset is randomly split into 100,000 training and 47,749 test samples. The length of the input sequence is set to 22 since the model uses the multivariate series representing the first and second lactation periods to predict the milk values (income) during the third lactation months (11 steps ahead). The Mean Absolute Error (MAE), Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) are selected to assess the sequence prediction performance of our model and the baselines. These error measures are calculated based on the difference between the predicted milk income (in Canadian dollars) and the ground truth values over the upcoming 11 months.

We compared our Transformer-BEATS model to eight other time series prediction and sequence modeling approaches, including Auto-ARIMA [3], MuMu [4], Vanilla Transformer [16], N-BEATS [12], Crossformer [22], DLinear [20], PatchTST [11] and iTransformer [9]. More details about these baseline models can be found in Related Work section.

Main Results

Table 1 represents the experimental results related to the dairy prediction, showcasing various error metrics computed across the target 11-month lactation period. The results highlight the superior performance of our Transformer-based model, surpassing other models, particularly outperforming all variations of Transformer models and the recently proposed linear model (DLinear). Here also, the proposed Transformer model obtained more similar predictions to the ground truth, compared to the other methods, specifically compared with the PatchTST model. The reason is that PatchTST ignores the intricate relations between the variables and the interactions among them. While our Transformer-BEATS model, similar to the iTransformer, takes those relations into consideration. Furthermore, the iTransformer model, which has been recently proposed, neglects the temporal dimension of time-series data, falling short in capturing long-term temporal correlations. The prediction results and visualization of the learned attention maps (both temporal and spatial attentions) show the advantage of combining the long-term temporal correlations with the inter-series patterns extracted from different encoder layers. By focusing on relevant time steps, the model can better understand the temporal dependencies and patterns present in the data, and the multiple layers of the inter-series encoders obtain spatial relations between the variables at different levels. In order to further analyse the effectiveness of our Transformer-BEATS, we divided the test sample series into tightly correlated and low correlated samples using the average Dynamic Time Warping (DTW) distance between their associated variables. Our model gains RMSE error improvements of 5% over the PatchTST model which doesn't deploy a mechanism to capture the interactions between the dairy variables. The prediction result of our model is also competitive and even better that those of the Crossformer and iTransformer models which exploit attention over the channel dimension to represent inter-series correlations. In addition, the results in terms of the error metrics in Table 1 pinpoint the promising performance of our model in dairy prediction of the sample dairy series with inter-variable correlations. Certain milk quality factors exhibit higher correlations, notably milk, fat, and protein yield [14]. Through our model's analysis, these dependencies are effectively highlighted as evidenced by the higher attention scores assigned to the corresponding feature pairs (milk, fat, and protein yields) in the inter-series attention map learned by our model. High attention scores are also assigned to DIM (Days in milk)-Lactose and DIM-FTP (Fat to Protein ratio) feature pairs, because the FTP and lactose amount are usually high during the early lactation months and as the lactation period progresses and the cow moves further into mid and late

Table 1. Comparison of the baseline methods with the proposed approach on the dairy production dataset in terms of cumulative Absolute Error and cumulative Root Mean Square Error, Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE).

Category	Model	RMSE of cumulative milk	MAE of cumulative milk	RMSE	MAE	MAPE
Univariate	ARIMA	52.675	42.342	7.313	5.769	27.197
	N-BEATS	32.212	24.974	4.198	3.257	17.479
Multivariate	Transformer	30.183	23.417	4.033	3.126	16.553
	MuMu	30.014	23.276	4.016	3.110	16.502
	DLinear	31.451	24.489	4.145	3.221	17.928
	Crossformer	30.039	23.288	4.017	3.111	16.454
	PatchTST	32.452	25.333	4.213	3.283	17.411
	iTransformer	30.564	23.718	4.069	3.156	16.772
	Transformer-BEATS	**29.947**	**23.208**	**4.004**	**3.100**	**16.448**

Table 2. Results of the ablation study (We run all the experiments three times with different seeds to reduce the randomness effect). The best results are indicated in bold text and the second best ones in italics.

Transformer components	Masked Attention	RMSE	MAE	Cumulative RMSE	Cumulative MAE
Backcast + Learned Weights	Yes	*4.004*	**3.099**	*29.952*	*23.205*
Backcast + Learned Weights	-	4.005	3.102	29.984	23.244
Inter-Series + Backcast + Learned Weights	Yes	**4.003**	*3.100*	**29.947**	23.208
Inter-Series + Backcast + Learned Weights	-	4.009	3.105	30.022	23.273
Inter-Series + Learned Weights	Yes	4.007	3.104	29.976	23.232
Inter-Series + Learned Weights	-	4.005	3.100	**29.947**	**23.189**
Inter-Series + Backcast + Fixed Weights	Yes	4.014	3.109	30.038	23.289
Inter-Series + Backcast + Fixed Weights	-	4.012	3.108	30.045	23.282

lactation, the lactose and fat to protein ratio tends to decrease [6].

Discussion

We further evaluate the spatial (inter-series) relations by inspecting the attention scores through visualization. Figure 3 shows the mean spatial attention map for the test samples (the learned attention map after training the model) together with the mean dynamic time warping (DTW) distance across the time steps of different channels. Three different blocks are selected in both attention map and the DTW distance matrix. The mean raw time series of the corresponding channels are plotted in Figure 2 to show the relation between the spatial attention scores and the similarity between the corresponding raw series (trend and shape).

According to Figure 3 (left figure) a1 block has a relatively high spatial attention score, which means PT-Milk Yield have similar trend and shape. Looking at Figure 2 top, we can observe that PT (Protein yield) and Milk Yield series are very similar in terms of the trend and shapelet. Meanwhile, blocks a2 and a3 indicate very small attention scores, and the middle and bottom plots in Figure 2 are the most clear evidence, by showing significantly different trends associated with the DIM and Milk Yield series (bottom sub-figure) and very different shapes between DIM and SCC (Somatic Cell Count) series (middle subfigure). Note that DTW distance matrix (Figure 3 right) represents very small distance between PT-Milk Yield (block b1), proving that the learned spatial attention scores could capture those types of similarities among the dairy series. Based on the DTW matrix (Figure 3 right), it's clear that the distance between DIM-SCC series (block b2) and DIM-Milk Yield (block b3) exhibit relatively higher values. Upon inspecting the associated attention map (blocks a2 and a3), it becomes apparent that their associated learned attention scores are notably low, as indicated by comparing their related series plots. This implies that our model has effectively discerned the disparity in trends between the DIM and Milk Yield series and the difference in shapes between DIM and SCC series according to the insights provided by the DTW measures. We derived the weight vector [0.0345, 0.1198, 0.6469, 0.1988] learned through the training of our model. These weights are assigned to the predictions of the first and second temporal encoders, as well as the first and second spatial encoders within the model, respectively. This allocation highlights the significant contribution of the first spatial encoder block (weight = 0.6469), emphasizing its crucial role in cross-channel representation.

Running Time and Performance Analysis

In this subsection, the training time of each model in one epoch is calculated and plotted against its corresponding error metric (Figure 4). Bases on this figure, our Transformer-BEATS model has the lowest error but its running time is

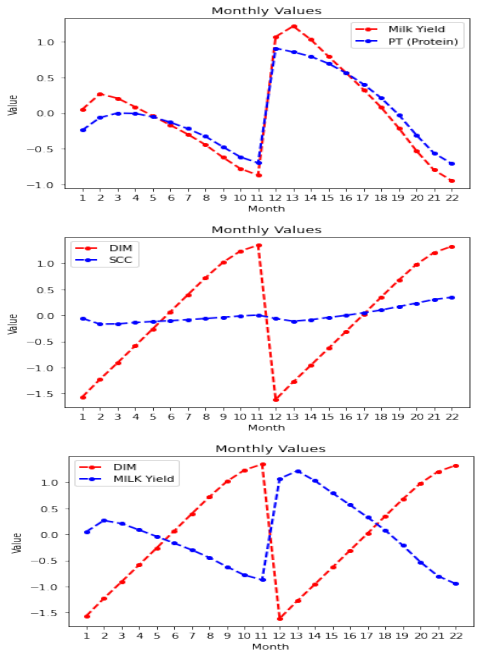


Fig. 2. Average normalized time series (Top: Mean Protein and milk yield, Middle: Mean DIM (Days In Milk) and SCC (Somatic Cell Count), Bottom: Mean DIM and milk yield).

higher than recurrent neural network based model (MuMu) and iTransformer. Our model is much faster than other Transformer based models recently proposed for time-series forcasting (PatchTST and Crossformer) while providing more accurate results in the task of dairy series forecasting.

Ablation Study

We conducted an ablation study on the designated dairy dataset to evaluate the efficacy of the different components incorporated in our proposed model. Each configuration was executed three times, and the averaged results are presented in Table 2. The ensuing observations provide insights into:

- Employing masking within the self-attention module, a technique applied in the basic Transformer model to ensure that predictions for individual positions solely rely on the preceding positions (causality), leads to slightly better prediction performance in most cases.
- Removing the inter-series self-attention encoders increases the cumulative RMSE error of the target lactation period, showcasing the importance of the inter-series encoders.
- Utilizing back-cast connections combined with the causal masking yields better outcomes compared to their absence.
- The learned weights enable the model to generate predictions by assigning significance to various outcomes in a data-driven manner and the learned weights are interpretable by showing the contribution of each component (encoder block). After establishing these weights (to fixed weights of 0.25), we conducted a replicated experiment to observe the influence of these adaptive weight assignments. The result indicates that our model with learnable weights, outperforms the same architecture set by the fixed weights.

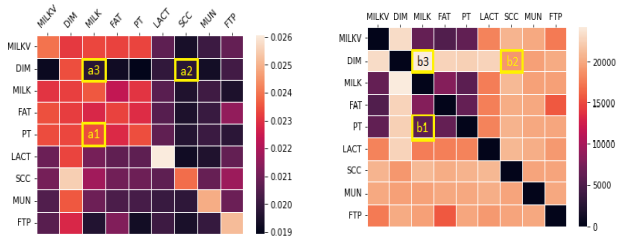


Fig. 3. Left: mean inter-series (spatial) attention map learned from the data samples (Numerical features), Right: mean channel-wise dynamic time warping distance matrix of the test samples. Blocks b1, b2 and b3 indicate a low, high and relatively high DTW distance between the dairy feature pairs. At the same time, blocks a1, a2 and a3 represent high, low and low attention scores, respectively. Therefore, a low DTW distance reflects a high attention score and vice versa.

Ablation on Longer Horizon Prediction and Other domains

We selected some public datasets from various domains, including traffic, weather, solar-energy and electricity for long term forecasting to conduct experiments using our Transformer-BEATS and the other baseline models. The results pinpoint the efficiency of our model when applied to other datasets and use cases according to Table 3. We should mention that our model is based on channel-mixing, which embeds all of the channels to the model dimension. In this experiments, the input series is divided into patches of equal length with a specific patch size and stride (overlapping region between two consecutive patches). Then, the patches are mapped into the model dimension and after adding the positional encoding are passed to the temporal encoder of the Transformer-BEATS model and the spatial encoder receives the transposed version of the original series without patching applied on it.

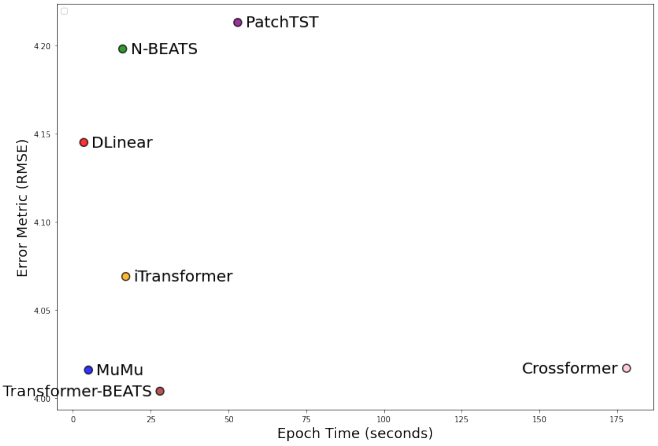


Fig. 4. Relationship between training epoch time (in seconds) and error metric (RMSE) for each model.

Additional experiments were conducted to indicate the performance of our model on different forecast horizons. The models were trained using a shorter input window and tested on longer prediction horizon (two lactation periods instead of one). Transformer-BEATS model achieved the best results in terms of the error metrics in comparison to the other baseline models

Table 3. Time series forecasting results for input length = 96 and prediction length = 12, 24, 48, 96 for PEMS07 dataset and prediction length = 96, 192, 336, 720 for others datasets. Best results are indicated in bold and second best results in italic.

Models		Transformer-BEATS		iTransformer		PatchTST		Crossformer		FEDformer		Autoformer	
Dataset	Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ECL	96	*0.153*	*0.256*	**0.148**	**0.240**	0.195	0.285	0.219	0.314	0.193	0.308	0.201	0.317
	192	*0.172*	*0.270*	**0.162**	**0.253**	0.199	0.289	0.231	0.322	0.201	0.315	0.222	0.334
	336	*0.193*	0.289	**0.178**	**0.269**	0.215	0.305	0.246	0.337	0.214	0.329	0.231	0.338
	720	*0.227*	**0.314**	**0.225**	*0.317*	0.256	0.337	0.280	0.263	0.246	0.355	0.254	0.361
PEMS07	12	**0.059**	**0.155**	*0.067*	*0.165*	0.095	0.207	0.094	0.200	0.109	0.225	0.199	0.336
	24	**0.083**	**0.183**	0.088	0.190	0.150	0.262	0.139	0.247	0.125	0.244	0.323	0.420
	48	*0.120*	*0.226*	**0.110**	**0.215**	0.253	0.340	0.311	0.369	0.165	0.288	0.390	0.470
	96	*0.181*	*0.283*	**0.139**	**0.245**	0.346	0.404	0.396	0.442	0.262	0.376	0.554	0.578
Weather	96	*0.164*	**0.214**	0.174	0.214	0.177	0.218	**0.158**	0.230	0.217	0.296	0.266	0.336
	192	*0.215*	0.260	0.221	**0.254**	0.225	*0.259*	**0.206**	0.277	0.276	0.336	0.307	0.367
	336	**0.271**	*0.297*	0.278	**0.296**	0.278	0.297	*0.272*	0.335	0.339	0.380	0.359	0.395
	720	0.382	0.366	0.358	*0.349*	*0.354*	**0.348**	**0.351**	0.386	0.403	0.428	0.419	0.428
Solar-Energy	96	**0.201**	*0.249*	*0.203*	**0.237**	0.234	0.286	0.310	0.331	0.242	0.342	0.884	0.711
	192	**0.233**	*0.271*	*0.233*	**0.261**	0.267	0.310	0.734	0.725	0.285	0.380	0.834	0.692
	336	*0.250*	*0.284*	**0.248**	**0.273**	0.290	0.315	0.750	0.735	0.282	0.376	0.941	0.723
	720	*0.273*	*0.309*	**0.249**	**0.275**	0.289	0.317	0.769	0.765	0.357	0.427	0.882	0.717

Table 4. Forecasting results related to longer horizon prediction in the dairy production dataset (Input sequence length = 11 months and target sequence length = 22 months).

Method	Cum-RMSE	Cum-MAE
Transformer-BEATS	**54.417**	**42.716**
DLinear	56.978	44.783
MuMu	55.024	43.198
iTransformer	57.118	44.814
Crossformer	54.623	42.780
PatchTST	57.613	45.666

according to Table 4. Our model predicts the milk income with 4.7% and 5.5% of Cumulative RMSE reduction compared to the iTransformer and PatchTST models, respectively, indicating its capability in handling longer horizon prediction of dairy production.

Conclusion

We propose Transformer-BEATS, a Transformer based framework for prediction of the multivariate time series of the dairy milk income in upcoming lactation months. Our model captures both temporal and cross-channel dependencies in the dairy time-series data by combining different Transformer encoders, each focusing on a specific task of capturing temporal or inter-series relations. Further, those module are inter-connected in a residual (back-cast) manner to help useful signal propagation to the next block. Our experiments on a large benchmark dairy dataset have shown that our proposed model outperforms several classical and state-of-the-art models.

Appendix: Dairy Dataset and Pre-processing

The benchmark dairy dataset used to evaluate our multivariate time-series model and the selected baseline methods consists in a set of dairy attributes that include metrics of milk quality, seasonality, year, health, and management factors, recorded during the first, second and third lactation cycles for 147,749 dairy cows from 5,844 Canadian dairy farms during the years of 2006 to 2017. More precisely, the dataset includes the following features: 1- HR-24-Milk (Milk yield in 24 hours), 2- Milk Value or Income (in CAD), 3- Somatic Cell Count (SCC as a health indicator), 4- Milk Urea Nitrogen (MUN is a health factor), 5- LACTOSE (Lactose yield), 6- FAT (Fat yield in 24 hours), 7- PT (Protein yield in 24 hours), 8- DIM (Days in Milk), 9- FTP (Fat to Protein ratio), 10- Lactation number, 11- MILK-FQCY (Number of milkings per day), 12- MILK-PTRN (Milking time in A.M, P.M, both), 13- Test season, 14- Birth season, 15- Animal condition, 16- Test year, 17- Test month. The prediction target series (variable) is the monthly income from milk sales (Milk Value in Canadian dollar (CAD)) measured at each cow during their third lactation. More information regarding the dairy factors (variables) can be found in Table 5.

Appendix: Public Datasets

Experiments are conducted on 4 complex real-world datasets to evaluate the performance of the proposed Transformer-BEATS model including (1) Weather [17] composed of 21 meteorological factors collected every 10 minutes from the Weather Station of the Max Planck Biogeochemistry Institute in 2020. (2) ECL [17] consists of the hourly electricity consumption data of 321 clients. (3) Solar-Energy [5] records the solar power production of 137 PV plants in 2006, sampled every 10 minutes. (4) PEMS07 is recorded based on the public traffic network data in California collected by 5-minute windows. The details are shown in Table 6.

Dairy Dataset Pre-processing

Similar to [10], the following steps were taken to pre-process the benchmark dataset:

- Retaining only animals with test records for the first, second, and third lactations.
- Omitting records from the dry period, i.e. the months during which a cow does not produce milk, which typically occurs between lactation cycles, with a near-zero milk value (income) as a result.
- Excluding animals that left the herd before or during the third lactation.

Table 5. Summary of numerical dairy variables used in the framework.

Variable	Missing %	Min	Max	Mean	Std
Milk Yield (Kg/24h)	1.4	0.20	61.40	29.60	9.25
Milk Value or Income (in Canadian Dollars (CAD))	0.002	0.09	42.95	21.21	5.94
Somatic Cell Count (1000/mL)	17.62	1	1754	156.59	254.87
Milk Urea Nitrogen (MUN)	56.04	2.10	19.80	10.91	3.27
Lactose Yield (Kg)	37.63	0	6.30	4.55	0.22
Fat Yield (Kg/24h)	1.64	0	2.40	1.19	0.38
Protein Yield (Kg/24h)	1.64	0	1.94	1.00	0.30
Days in Milking (DIM)	0	0	517	169.56	99.39
Fat to Protein Ratio (FTP)	1.64	0.73	1.67	1.20	0.16

Table 6. Detailed dataset descriptions. Dim denotes the variate number of each dataset. Dataset Size denotes the total number of time points in (Train, Validation, Test) split, respectively. Frequency indicates the sampling interval of time points.

Dataset	Dim	Prediction Length	Dataset Size (Train, Validation, Test)	Frequency	Field
ECL	321	96, 192, 336, 512	(18317, 2633, 5261)	Hourly	Electricity
PEMS07	883	12, 24, 48, 96	(16911,5622,468)	5 min	Traffic
Weather	21	96, 192, 336, 512	(36792, 5271, 10540)	10 min	Weather
Solar Energy	137	96, 192, 336, 512	(36601, 5161, 10417)	10 min	Energy

- Removing duplicate records.
- Eliminating records with negative milk value and cumulative milk value: Although these cases had constituted a minor fraction of the dataset, they were removed to avoid inconsistencies. For instance, negative values of dairy income (expressed in CAD), may be the result of data acquisition errors or exogenous expenses such as the acquisition of new equipment or repairs.
- Deleting records with contradictory data, such as rows implying an active milking phase but featuring zero milk yield.
- Removal of outliers (their respective rows): Outlier records are those with more than $\pm$ 2.5 $\times$ Standard Deviation range for the mean of one (or more) dairy attribute.
- Missing data imputation: Using the herd ID, season and year (grouping the animals and using mean of each group to replace the missing values that lie in the same group).

From the 147,749 dairy cows, we randomly selected 100,000 to train the models and the remaining 47,749 for evaluation. The missing data was imputed after train-test split to avoid information leakage [10]. The categorical variables are transformed using one-hot encoding, while the numerical variables (series) are standardized by taking into account their mean and standard deviation across both batch and time dimensions. Then, they are merged as the final multivariate series (52 series in total).

Appendix: Experimental Setting

We used the Grid Search method for hyper-parameter optimization [23]. We investigated a set of possible values for each hyper-parameter and trained the model using it. For example, different model dimensions were employed, and the one leading to the lowest prediction error during the testing time was selected in the final model configuration. The same process is performed for other hyper-parameters, including learning rate, batch size, optimizer type and number of train epochs.

In experiments regarding public datasets, we trained the models using MSE (Mean Square Error loss) for 10 epochs using Adam optimizer and the early stopping policy (patience number set to 3). We divided the input series into patches of the same size (length of 8) according to [11] which reduces the computational cost and helps the model to capture the long-term temporal correlations. We should mention that Transformer-BEATS (ours) takes Channel-mixing in treating time series representation (it maps the whole channels to the model dimension).

References

1. MR Borchers, YM Chang, KL Proudfoot, BA Wadsworth, AE Stone, and JM Bewley. Machine-learning-based calving prediction from activity, lying, and ruminating behaviors in dairy cattle. *Journal of dairy science*, 100(7):5664–5674, 2017.
2. Jelle Bruinsma. *World agriculture: towards 2015/2030: an FAO study*. Routledge, 2017.
3. Javier Contreras, Rosario Espinola, Francisco J Nogales, and Antonio J Conejo. Arima models to predict next-day electricity prices. *IEEE transactions on power systems*, 18(3):1014–1020, 2003.
4. Charlotte Gonçalves Frasco, Maxime Radmacher, René Lacroix, Roger Cue, Petko Valtchev, Claude Robert, Mounir Boukadoum, Marc-André Sirard, and Abdoulaye Baniré Diallo. Towards an effective decision-making system based on cow profitability using deep learning. In *ICAART (2)*, pages 949–958, 2020.
5. Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 95–104, 2018.

6. Dong-Hyun Lim, Vijayakumar Mayakrishnan, Hyun-Jeong Lee, Kwang-Seok Ki, Tae-Il Kim, and Younghoon Kim. A comparative study on milk composition of jersey and holstein dairy cows during the early lactation. *Journal of Animal Science and Technology*, 62(4):565, 2020.
7. Arno Liseune, Matthieu Salamone, Dirk Van den Poel, Bonifacius Van Ranst, and Miel Hostens. Predicting the milk yield curve of dairy cows in the subsequent lactation period using deep learning. *Computers and Electronics in Agriculture*, 180:105904, 2021.
8. Minghao Liu, Shengqi Ren, Siyuan Ma, Jiahui Jiao, Yizhou Chen, Zhiguang Wang, and Wei Song. Gated transformer networks for multivariate time series classification. *arXiv preprint arXiv:2103.14438*, 2021.
9. Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.
10. Vahid Naghashi, Gabriel Dallago, Abdoulaye Banire Diallo, and Mounir Boukadoum. Univariate and multivariate time-series methods to forecast dairy income. In *2nd AAAI Workshop on AI for Agriculture and Food Systems*, 2023.
11. Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.
12. Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*, 2019.
13. Matthieu Salamone, Ines Adriaens, Andy Vervaet, Geert Opsomer, Hadi Atashi, Veerle Fievez, Ben Aernouts, and Miel Hostens. Prediction of first test day milk yield using historical records in dairy cows. *animal*, 16(11):100658, 2022.
14. LR Schaeffer and J Jamrozik. Multiple-trait prediction of lactation yields for dairy cows. *Journal of Dairy Science*, 79(11):2044–2055, 1996.
15. Sho Ushikubo, Chikara Kubota, and Hayato Ohwada. The early detection of subclinical ketosis in dairy cows using machine learning methods. In *Proceedings of the 9th International Conference on Machine Learning and Computing*, pages 38–42, 2017.
16. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
17. Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The eleventh international conference on learning representations*, 2022.
18. Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.
19. Neo Wu, Bradley Green, Xue Ben, and Shawn O'Banion. Deep transformer models for time series forecasting: The influenza prevalence case. *arXiv preprint arXiv:2001.08317*, 2020.
20. Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.
21. Guoqiang Zhang, B Eddy Patuwo, and Michael Y Hu. Forecasting with artificial neural networks:: The state of the art. *International journal of forecasting*, 14(1):35–62, 1998.
22. Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2022.
23. Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11106–11115, 2021.
24. Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning*, pages 27268–27286. PMLR, 2022.

scientific reports

Check for updates

OPEN

A multiscale model for multivariate time series forecasting

Vahid Naghashi[1,2], Mounir Boukadoum[1,2] & Abdoulaye Banire Diallo[1,2]✉

Transformer based models for time-series forecasting have shown promising performance and during the past few years different Transformer variants have been proposed in time-series forecasting domain. However, most of the existing methods, mainly represent the time-series from a single scale, making it challenging to capture various time granularities or ignore inter-series correlations between the series which might lead to inaccurate forecasts. In this paper, we address the above mentioned shortcomings and propose a Transformer based model which integrates multi-scale patch-wise temporal modeling and channel-wise representation. In the multi-scale temporal part, the input time-series is divided into patches of different resolutions to capture temporal correlations associated with various scales. The channel-wise encoder which comes after the temporal encoder, models the relations among the input series to capture the intricate interactions between them. In our framework, we further design a multi-step linear decoder to generate the final predictions for the purpose of reducing over-fitting and noise effects. Extensive experiments on seven real world datasets indicate that our model (MultiPatchFormer) achieves state-of-the-art results by surpassing other current baseline models in terms of error metrics and shows stronger generalizability.

Time series forecasting plays an essential role in many fields, including finance, agriculture, meteorology and energy consumption domains. Motivated by its widespread application in different fields of Natural Language Processing and Computer Vision[1,2], Transformer[3] is exploited in various time series applications (forecasting, imputation and classification) with some modifications in its architecture[4,5]. While recent studies have raised doubts about the performance of Transformers, specifically when employing linear models with superior performance[6], the inherent capabilities of Transformers remain promising[7,8], particularly in representing complex and non-linear datasets. This underscores the need for further modifications to fully harness their capability in the domain of time series forecasting.

Real-world multivariate time series exhibit high correlations between different variates and fluctuations at various temporal scales. For example, electricity consumption shows specific temporal variations spanning seasonal, daily and hourly granularities. Figure 1, illustrates a time series of a stock over one year, in which relations between patches of different scales are critical to capture more information regarding the local and global temporal dependencies from various perspectives. This calls for multi-scale modeling of time series[9] and representation of inter series correlations[10].

Most of the existing time series forecasting models lack an efficient mechanism for multi-scale representation and they totally rely on a single scale or time resolution. Although some of the baseline models consider multi-scale representation in their modeling process[8,11], however, these models employ separate sets of parameters for capturing temporal dependencies at each scale, which significantly increases time complexity and the risk of overfitting. Furthermore, most of the aforementioned methods, ignore the cross-channel relationships between time series channels which has been proved to be critical in time series analysis task[10]. Another limitation of the current research is the single step decoding of the encoded representation, particularly in dealing with long horizon prediction, which raises the risk of overfitting and makes the model more susceptible to be affected by noise.

To leverage the capabilities of Transformers for addressing the above mentioned issues, we aim to enhance the capture of both multi-scale and cross-channel dependencies, thereby aggregating this information for time series representation. We further divide input series into local patches to process the input time series and since the variable independence has been proved to be more efficient[7], we feed the multivariate series to the Transformer blocks by keeping the variate (channel) dimension intact. We further divide the series into patches of different size and map patch length to the model dimension by using 1-dimensional convolution in order to leverage the local information inside a patch (intra-patch relations). Instead of using several Transformer blocks, we first embed the time series using different scales and aggregate them into the same model dimension (feature space). Then, we process the mapped series using a Transformer block, followed by an channel-wise

[1]Computer Science, Université du Québec à Montréal, Montreal, Canada. [2] Vahid Naghashi, Mounir Boukadoum and Abdoulaye Banire Diallo contributed equally to this work. ✉email: diallo.abdoulaye@uqam.ca

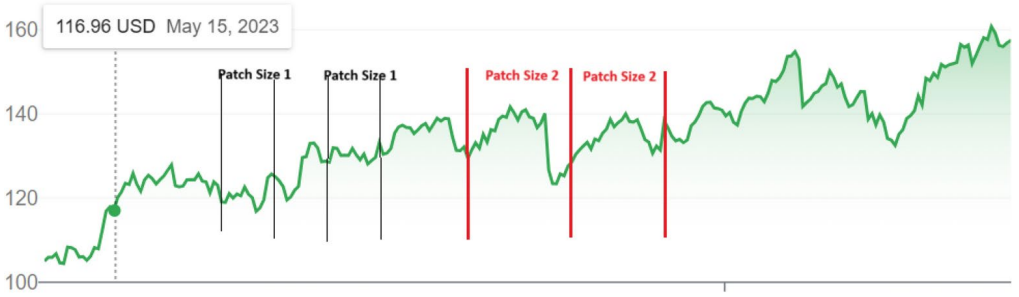


Fig. 1. Multi-scale dependencies in a real-world time series instance. The temporal patterns within patches of lengths Patch Size 1 and Patch Size 2 show similar trends and seasonality. Capturing these relations across different scales is crucial for analyzing time series data effectively.

Transformer component to capture cross-series dependencies. To further improve the Transformer for time series forecasting task, we introduce a channel dimensionality reduction method inside the multi-head self attention module when applied on the channel dimension, which helps to reduce over-fitting noise, specifically when dealing with the time series composed of many variates (channels). We additionally, devise a multiple step and pseudo auto-regressive decoding of prediction window by generating the predictions segment by segment using the encoded series and previously generated segment, in order to reduce the noise impact, particularly in long-term forecasting scenarios. The effectiveness of the proposed model is verified through different real-world benchmarks.

Here is the key summarization of our contributions:

1. We propose a time series forecasting model which effectively utilizes the multi-scale information and captures the inter-series dependencies among different channels.
2. Our model demonstrates superior performance on various time series datasets and shows robust performance in time series forecasting task with long input length.
3. We design a multi-scale time series embedding method which captures the local temporal information from various scales and divides time series to patches of different lengths, preparing it to represent long term temporal correlations from multiple temporal aspects.
4. We propose a pseudo auto-regressive (multi-step) decoding scheme to forecast the time series in multiple sequential steps rather than decoding the whole prediction length using a single linear layer in order to reduce the noise effect.

Related work

Time series forecasting involves predicting future values of one or multiple series based on the historical information. Time series forecasting methods are mainly categorized into classical and deep learning models. Among the statistical models, ARIMA[12] and methods based on exponential smoothing[13] are well-known baselines for time series forecasting. The deep learning methods are based on Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN) and Transformer models. Among the RNN based model, DeepAR exploits RNN and auto-regressive decoding to predict the time series[14]. CNN models use convolutions to extract temporal and sub-series dependencies[15,16]. For example, SCINet[16] utilizes multiple convolutions to extract temporal information from different down sampled versions of the series. TimesNet[15] on the other hand, converts the original one-dimensional time series into two-dimensional series and uses convolutions to capture inter-period and intra-period information. Some other linear models based on Multi-Layer Perceptron (MLP) have been proposed in[6,17] which exhibit an effective performance in time series domain. Several works have leveraged Graph Neural Networks (GNN) for time series forecasting, specifically for traffic forecasting. For example, MTGNN[18] utilizes temporal and graph convolutional layers to capture temporal and spatial (cross-channel) dependencies. An attention based spatial-temporal graph neural network is proposed in Ref.[19] which captures the temporal dynamics of traffic series by using the local context. In Ref.[20], a feature correlation-aware spatio-temporal graph convolutional network is designated for traffic prediction, which captures multi-scale spatial and temporal relations effectively, considering cross-scales dependencies. Dynamic graph structure learning for multivariate time series forecasting[21] exploits graph learning networks to capture hidden dependencies between variables, enhancing the accuracy of forecasting by effectively capturing complex interrelationships within the data. Another work[22] addresses the problem of capturing dynamic correlations by learning historical relation graphs and predicting future relation graphs. They also design a causal GNN for feature extraction and reasoning network to capture the relations between historical time steps and forecasting horizon. Recent studies in time series analysis have also focused on reducing the computational and memory requirements related to processing large datasets. For instance, Ref.[23] introduced TimeDC, a time series dataset condensation framework to preserve complex temporal relations while significantly reducing dataset size.

With the outstanding breakthrough in Natural Language Processing (NLP) and Computer Vision (CV) fields, Transformer models have recently shown superior performance in time series forecasting task and they have been continuously evolving. Among them, Informer[5] develops a Transformer model based on prob-sparse self-attention to select important keys and reduce time complexity of self-attention. In Autoformer[24],

the self-attention is replaced with auto-correlation to capture temporal dynamics. FEDformer[4] utilizes Fourier transformer to deal with time series data given the fact that time series tend to have a sparse representation in Fourier basis. Recently, some linear models have been developed which outperform Transformer models in time series domain[6] and raised the concern about the efficiency of Transformer for time series forecasting. However, the PatchTST model[7] proved the efficiency of Transformer models in time series analysis, which exploits patching and channel independence to model time series data, pinpointing that Transformer architecture is still a powerful model with some adaptation and architectural adjustments. Following PatchTST, other Transformer models have been developed for time series and proved high capability in dealing with high-dimensional time series[10]. Multi-scale representation proved to be necessary for time series analysis[11,25]. Triformer[25] designs a patch attention with linear complexity and variable specific parameters to enhance accuracy. Reference[11] develops a multi-scale framework to model time series using different resolutions and they utilize separate predictive models for each temporal scale which leads to high computational complexity. After the rise of generative pre-trained models, Large Language Models (LLMs) have been utilized for time series forecasting by fine-tuning and exploiting prompt engineering, e.g., Time-LLM[26] which keeps the backbone LLM intact and input time series is reprogrammed with text prototypes before feeding it to the frozen LLM, aligning two modalities. GPT4TS[27] employs GPT-2[28] model for time series forecasting by feeding the time series patches to the model in a Channel-Independent manner. A spatial-temporal large language model is proposed for traffic forecasting[29] by defining spatial-temporal embedding to learn the spatial locations and global temporal dependencies of time steps at each location. In traffic prediction often capturing spatial-temporal dependencies at multiple scales is required. To address the mentioned requirement, MT-STNets is designed in[30], for prediction of both fine-grained traffic conditions on individual roads and coarse-grained traffic flows across urban areas. More recently, Pathformer[8] is proposed which exploits adaptive pathways to capture multi-scale temporal relations in an adaptive manner by automatically selecting patches of different resolutions, which uses separate set of parameters for each temporal granularity in its design. Different from the above mentioned models, which either disregard multi-scale modeling entirely or represent multi-scale information inside their model architecture, we devise a way to project multiple-scale temporal representations to the model dimension in the time series embedding phase and utilize the same model parameters to capture multi-scale and cross-scale information efficiently.

We expect improvements by using multi-scale embedding, since real-world time series often exhibit multiple seasonal patterns and modeling all the scales would improve the model performance. We employed Fourier analysis to calculate the high frequency components and dominant periods of time series samples from various benchmark datasets in order to show that real-world datasets usually rely on more than one seasonality pattern (scale). For example, electricity dataset typically exhibits daily (e.g., 24-h or 48-h) seasonality, together with long-term periods, such as monthly consumption or seasonal peaks (with periods of 90, 102). Multi-scale embedding and sharing the same model space among different scales is the main difference between our proposed method and the existing literature which exploits the interactions among different scales. We also use channel dimensionality reduction through 1-dimensional convolution over the channel dimension in the cross-channel encoder layer, which helps to reduce the noise effect in the highly correlated time series by mitigating the over-fitting chance. The previous methods usually utilize a single linear layer to map the model dimension to the prediction window, which would lead to over-fitting in longer horizon forecasting scenarios. Therefore, we design a simple but effective way to avoid this effect, by decoding the extracted information through linear layers over consecutive steps.

Methods

In this paper, we deal with the problem of time series forecasting which can be stated as following: Given a set of historical time series of C variates over L timestamps $(X_1, X_2, ..., X_L)$, with X_i indicating the observation at timestamp i of dimension C, we aim to predict F steps of the future time steps: $(X_{L+1}, ..., X_{L+F})$. To effectively capture multi-scale information and inter-series correlations, we propose MultiPatchFormer, which utilizes a novel predictor composed of a sequence of linear layers to simulate auto-regressive decoding in patch level. The overall architecture is illustrated in Fig. 2. The whole framework consists of instance normalization[31], multi-scale embedding, temporal encoder, inter-series encoder module and predictor. Instance normalization[31] is a technique employed by many models to avoid distribution shifts between training and test sets. The multi-scale embedding first divides the input series into patches of various sizes and then embeds all of them to the same model space. In other words, given 4 temporal scales, time series is reshaped to $((B, C), L, 1)$, then is convolved with filters of different sizes and strides, corresponding to the patch sizes and strides (overlapping region between to consecutive patches). B, C and L indicate batch size, channels and series length, respectively. We deliberately decided to exploit one-dimensional convolution to split series for the purpose of capturing local temporal dynamics inside a patch (intra-patch relations). After splitting the input time series by using patches of four various lengths, they are merged together into the model dimension (d_{model}). To be more specific, the output channel of the individual convolutions are set to $\frac{d_{model}}{4}$, mapping each patched series into its own smaller sub-space and then merging them to the main model dimension. This strategy employs separate 1-dimensional convolutions with input channel of 1 and output channel of $\frac{d_{model}}{4}$ to project multi-scale information into the embedded space and each channel of time series is embedded independently in this manner, as we reshape the series into $((B, C), L, 1)$ before applying the convolutions. In order to reduce the over-fitting effect in channel-wise multi-head self-attention, we devise a simple but yet effective solution for reducing noise and time complexity of time series with large number of channels. The regular linear predictor which is usually used as the final layer of time-series forecasting models, is susceptible to over-fitting, specifically when the forecast horizon is relatively long. Rather than using a simple linear layer to map from model dimension to the prediction length, we break the final linear layer to multiple linear layers to decode the predictions in multiple steps by generating part of

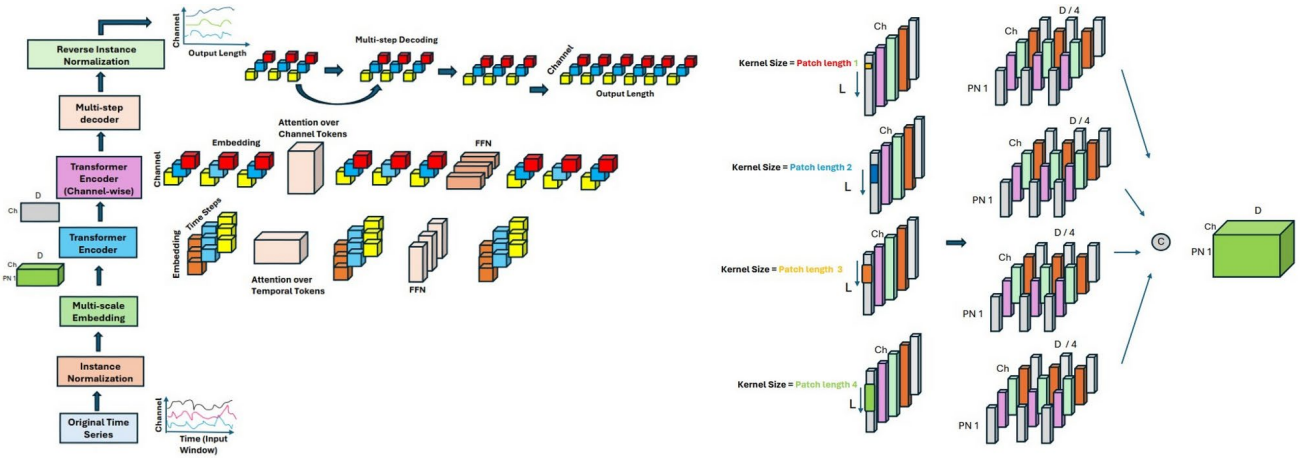


Fig. 2. General overview of the proposed framework (MultiPatchFormer). Left: the main steps include normalization, multi-scale embedding using different patch sizes, temporal encoder, reshaping the output of the temporal encoder using a 1-dimensional convolution layer and sending the result to the channel-wise encoder. The output of the channel-wise encoder is passed through the multi-step decoder to generate the final predictions followed by the de-normalization step. Right: Multi-scale embedding is illustrated using 4 different patch sizes (scales), where *D* refers to the model dimension (d_{model}), *Ch* indicates time series channels (variates), and *PN*1 refers to the number of patches fixed across four scales by choosing appropriate strides.

the prediction horizon at each step. In the following section, each component of our model is described in more detail.

Multi-scale embedding

In Computer Vision, prior to the main backbone, RGB channels are embedded into a D-dimensional vector at every pixel. This embedding process serves to mix information from channels through the embedding layer. However, employing a similar variable-mixing embedding, such as embedding *M* variables into a D-dimensional vector per time step, proved to be unsuitable for time series data because of two primary reasons. Firstly, the difference between variables within a time series is far greater than that among RGB channels within an image[25]. A mere embedding layer is insufficient for capturing the intricate correlations among variables, which might lead to the loss of their individual behavior. We refer X_{in} as the *C* variables input time series of length *L* which is further divided into *PN* patches of patch length *PL* after proper padding (padding by repeating the last time element stride times). The stride *S* is specified as the length of non overlapping region between two consecutive patches. Then, the patches will be embedded into embedding vectors of dimension $\frac{d_{model}}{num_{scales}}$:

$$X_{emb} = Embed_s(X_{in}) \tag{1}$$

In which, $X_{emb} \in C \times PN \times \frac{d_{model}}{num_{scales}}$ is the input embedding for one temporal scale and $Embed_s$ denotes embedding with scale or patch length of *s*. As we mentioned earlier we conduct this patchify embedding in a fully-convolution way to simplify computation and capture local dynamics in various scales. First, we extend (unsqueeze) the shape to $X_{emb} \in C \times L \times 1$, and then we feed the padded X_{embed} into a 1-dimensional convolution layer with kernel size *PL* and stride *S*, which maps the input with one channel into $\frac{d_{model}}{num_{scales}}$ output channels. It should be mentioned that in above process, each of the *C* univariate time series is embedded independently which keeps the variate dimension intact and helps to model inter-variable dependencies in the following blocks. We repeat the above embedding process for num_{scales} times, each time with different patch size and stride to capture and embed different scales of the input series. We utilize appropriate stride sizes with each patch length to obtain the same number of patches (patch numbers), because the resultant embeddings are concatenated as the final representation. Therefore, The multi-scale embedding is the result of concatenating embeddings with various patch sizes which are combined along their model dimension:

$$X_{emb} = Concatenate(Embed_1(X_{in}), ..., Embed_{num_{scale}}(X_{in})) \tag{2}$$

In our model, the embeddings of multiple scales are projected into a shared feature space. For example, given 4 different patch sizes, we project the time series into the same feature space by dividing the model dimension into 4 segments and assigning the embedding result of each scale to one segment of the model dimension. In other words, each scale is first projected into $model_{dim}/4$ space and then these embeddings are concatenated along the model dimension. This allows for interaction between different temporal resolutions within the same representation space and capturing multi-scale and cross-scale information through the self-attention component.

Temporal encoder

We define a set of different patch sizes $(P_1, P_2, ..., P_M)$ to represent various views corresponding to temporal resolution of input series. Each divided patch is embedded to a lower dimension than the main model dimension (to $\frac{d_{model}}{4}$ in case of using 4 different scales), which are then concatenated to produce the final embedding ready to be fed into the temporal and channel-wise encoders, respectively. The embedded series is first passed through the temporal encoder, which consists of a multi-head self attention followed by layer normalization and feed-forward layers to capture long-term temporal dependencies across multiple scales, simultaneously. The multi-head self-attention in temporal encoder, extracts temporal dependencies over the patches. Consider a set of divided patches which are the result of multiple scale division: $(P_1, P_2, ..., P_{PN})$, with *PN* indicating number of patches fixed for all scales and $P_i \in R^{d_{model}}$. Then, we employ trainable linear transformations to obtain query, key and value matrices, $Q, K, V \in R^{PN \times d}$ and perform attention score calculation as:

$$AttentionScore = softmax\left(\frac{Q \cdot K^T}{\sqrt{d}}\right) \tag{3}$$

In the above formula, *d* indicates the head dimension $\frac{d_{model}}{h}$, with *h* showing the number of heads. The attention scores are multiplied by the value matrix to obtain the result of multi-head self-attention over divided patches which demonstrates temporal dependencies along the patches.

Channel-wise attention

The output of the temporal encoder is passed to the channel-wise encoder, which employs a cross-channel multi-head attention module to capture the correlations among the channels (variates). However, the shape of resultant tensor of temporal encoder is $(B \times C) \times PN \times d_{model}$, which is reshaped into $B \times C \times (PN \times d_{model})$. Then we exploit a 1-dimensional convolution to reduce the last dimension into d_{model} (model dimension), preparing it to be fed into the channel-wise encoder layer. Therefore, the input of the channel-wise multi-head attention is of shape $B \times C \times d_{model}$. When performing attention over channel dimension, the vanilla self-attention may suffer from high computation and over-fitting, specially when dealing with a high-dimensional dataset with many variates. To reduce the noise over-fitting issue and computational complexity, we devise a solution to summarize the key and value matrices in channel-wise attention step. In other words, query matrix remains intact of shape $B \times C \times d$ and we apply a 1-dimensional convolution along the channel dimension with the same kernel size and stride (with proper padding) to reduce the number of channels in key and value matrices, resulting in tensors of size $B \times r \times d$ with $r < C$ which indicates the summarized channels. We deploy kernel and stride size according to the dataset in out experiments, e.g. kernel size and stride of 21 with padding 10 on each side of the input series are utilized for Traffic dataset. Therefore, it leads to decrease in time complexity and noise effect when calculating attention scores over the channel dimension:

$$AttentionScore = softmax\left(\frac{Q_{channel} \cdot K_{channel}^T}{\sqrt{d}}\right) \tag{4}$$

In which $Q_{channel} \cdot K_{channel}^T$ requires $C \times d \times r$ multiplications rather than $C^2 \times d$. The resultant attention output is further passed through layer normalization and feed-forward layers to extract meaningful features considering the dependencies over the channel dimension.

Multi-step decoder

Both long-term temporal and channel-wise dependencies are captured through the employed encoders in previous steps. The final output which comes from the encoder over channel dimension is of dimension $B \times C \times d_{model}$. Typically, a linear layer is employed as the final decoder or predictor to map from model dimension to prediction length (forecast window). But, the number of parameters in the output linear layer increases when dealing with long prediction horizons which might lead to over-fitting and affecting the far-future predictions by noise. To mitigate this issue, we propose a multi-step decoding process in which a sequence of linear layers are applied to the encoded feature map from the channel-wise encoder and current prediction results. In other words, we first divide the prediction length into different parts (length of 4 or 8) and generate the prediction part by part, by applying the combination of the final feature map and previously predicted parts to a linear layer. The multi-step decoding process is illustrated in Fig. 3.

Results

Datasets and baselines

Datasets We conduct different experiments using 7 real world datasets including electricity Transformer Temperature[24], weather forecasting[24], traffic[24] and electricity consumption[24]. These datasets consists of the well-known ETT (ETTh1, ETTh2, ETTm1, ETTm2), Weather, Electricity and Traffic. More details about the datasets are reported in Table 1. **Baseline and metrics** 14 well-known time series forecasting models are selected as baselines, including Pathformer[8], PatchTST[7], DLinear[6], NLinear[6], Crossformer[32], Scaleformer[11], TIDE[17], FEDformer[4], Pyraformer[33], Autoformer[24], Time-LLM[26], GPT4TS[27], MTGNN[18] and SDGL[21]. To ensure fairness in our experiments, the input length is fixed for all models (L = 96) and prediction length consists of (F = 96, 192, 336, 720). Since the authors of Time-LLM used input length of 512 in their paper[26], we set the context window to 512 in LLM comparison experiments and trained our model on different datasets with input length of 512 and prediction length in {96, 192, 336, 512}. We compare our model with graph models based on input window of length 96. Two well-known error metrics in time series forecasting are selected: Mean Absolute Error (MAE)

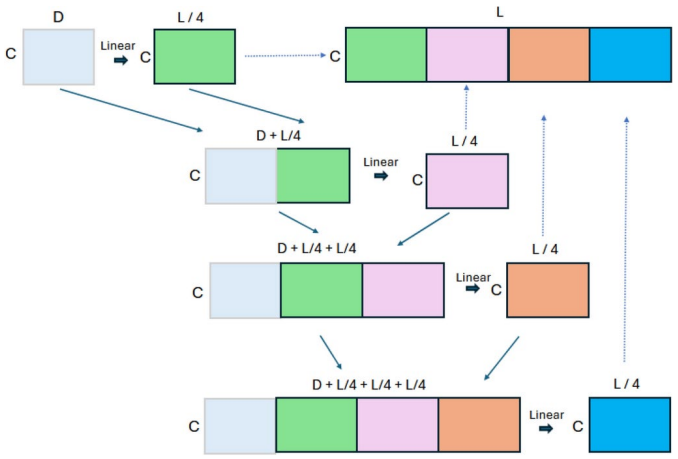


Fig. 3. Multi-step decoder is proposed to reduce noise effect in long prediction length. *C*, *D* and *L* refer to channels, model dimension and prediction length, respectively. At each step the feature map from the channel-wise encoder is concatenated with previously generated prediction segments and passed through a linear layer to generate the next prediction segment (part). The final prediction is the concatenation result of the individual predicted segments.

Dataset	Dim	Pred length	Dataset size	Mean corr	Max corr	Trend	Freq	Field
Electricity	321	96, 192, 336, 720	(18,317, 2633, 5261)	0.981	0.999	0.315	Hourly	Electricity
ETTh1, ETTh2	7	96, 192, 336, 720	(8545, 2881, 2881)	0.551 (0.551)	0.897 (0.946)	0.797 (0.761)	Hourly	Electricity
ETTm1, ETTm2	7	96, 192, 336, 720	(34,465, 11,521, 11,521)	0.551 (0.551)	0.897 (0.945)	0.896 (0.822)	15 min	Electricity
Weather	21	96, 192, 336, 720	(36,792, 5271, 10,540)	0.704	0.999	0.039	10 min	Weather
Traffic	862	96, 192, 336, 720	(12,185, 1757, 3509)	0.751	0.985	0.031	Hourly	Transportation

Table 1. Detailed dataset descriptions. Dim indicates the variate number of each dataset. Dataset Size denotes the total number of time points in (Train, Validation, Test) splits, respectively. Mean and maximum cross-correlations (Dynamic Time Warping based) between the channels of each dataset are also reported in Mean Corr and Max Corr columns. Trend column indicates strength of trend for each dataset.

and Mean Square Error (MSE) to compare the models. **Implementation details** We utilize Adam optimizer with learning rate in $\{10^{-3}, 10^{-4}\}$ and L1 loss function. The models are trained for 10 epochs with early stopping (patience of 3) based on the validation loss. We implemented our model using Pytorch framework[34] and the experiments are executed on an NVIDIA A100 40GB GPU. MultiPatchFormer utilizes 4 different scales to patchify and embed input time series (common scales in the time series forecasting domain, e.g., 8, 16, 24, 48 with proper strides, 8, 8, 7, 6 to create the same number of patches).

Main results

The main results are summarized in Tables 2, 3 and 4. The results related to the baseline models are taken from Ref.[8] and we follow the same settings to train and evaluate our model. All experiments are repeated 5 times and the average results are reported. According to Table 2, our MultiPatchFormer, shows better performance across four different prediction horizons. Regarding error measures across four prediction lengths, our model achieves the best or second best performance among the state-of-the-art models in 82% of cases in MSE metric and 86% of the conducted experiments in terms of MAE measure. Our MultiPatchFormer outperforms the baseline models on benchmarks with a high number of variates and complex structure. For instance, in Traffic dataset (862 covariates), MultiPatchFormer persistently outperforms the second best baseline by more than 5% on average MSE and 7.7% on average MAE across four prediction windows, while consuming less training time and parameters. By exploiting 321 variates in ECL (Electricity) dataset, we achieved average error reduction of 3.7% compared to Pathformer[8] and 10.7% improvement over the PatchTST model. This highlights that MultiPatchFormer is capable of utilizing extensive covariate dependencies for high accuracy prediction. Remarkably, our model makes predictions with lower error in complex problems, including Electricity (ECL) and Traffic, which underscores the efficiency of the channel-wise attention and multi-scale embedding to capture the inter-series dependencies across various scales. It worth noting that while NLinear[6] achieves promising performance in some datasets, MultiPatchFormer significantly outperforms the MLP-based models (NLinear and DLinear) by over 26.8% error reduction on Traffic and more than 10% improvement on Electricity dataset. This indicates that Transformer models are still powerful in forecasting long-term time series tasks. In terms of cross-variate modeling, Crossformer[32], utilizes a variant of attention to capture those type of correlations, but our model gains impressive improvement of over 28% and 18% compared to the Crossformer model on

Models	Ours		Pathformer		PatchTST		DLinear		NLinear		Crossformer		Scaleformer		TIDE		FEDformer		Autoformer	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1																				
96	**0.315**	**0.342**	*0.316*	*0.346*	0.324	0.361	0.392	0.405	0.386	0.392	0.429	0.440	0.396	0.440	0.356	0.381	0.379	0.419	0.505	0.475
192	0.367	**0.369**	*0.366*	*0.370*	**0.362**	0.383	0.441	0.436	0.440	0.430	0.494	0.482	0.434	0.460	0.391	0.399	0.426	0.441	0.553	0.496
336	0.399	**0.394**	**0.386**	**0.394**	*0.390*	*0.402*	0.501	0.478	0.480	0.443	0.706	0.625	0.462	0.476	0.424	0.423	0.445	0.459	0.621	0.537
720	0.464	**0.432**	**0.460**	**0.432**	*0.461*	*0.438*	0.538	0.526	0.486	0.472	0.750	0.689	0.494	0.500	0.480	0.465	0.543	0.490	0.671	0.561
Avg	0.386	**0.384**	**0.383**	*0.386*	*0.384*	0.396	0.468	0.461	0.448	0.434	0.595	0.559	0.447	0.469	0.413	0.417	0.448	0.452	0.588	0.517
ETTm2																				
96	*0.171*	*0.250*	**0.170**	**0.248**	0.177	0.260	0.186	0.279	0.177	0.257	0.197	0.321	0.182	0.275	0.182	0.264	0.203	0.287	0.255	0.339
192	**0.236**	**0.294**	*0.238*	*0.295*	0.248	0.306	0.266	0.339	0.241	0.297	0.326	0.375	0.251	0.318	0.256	0.323	0.269	0.328	0.281	0.340
336	*0.303*	*0.337*	**0.293**	**0.331**	0.304	0.342	0.332	0.376	0.302	0.337	0.372	0.421	0.340	0.375	0.313	0.354	0.325	0.366	0.339	0.372
720	*0.397*	0.393	**0.390**	**0.389**	0.403	0.397	0.462	0.455	0.405	*0.396*	0.410	0.448	0.435	0.433	0.419	0.410	0.421	0.415	0.433	0.432
Avg	*0.277*	*0.319*	**0.273**	**0.316**	0.283	0.326	0.312	0.362	0.281	0.322	0.326	0.391	0.302	0.350	0.293	0.338	0.305	0.349	0.327	0.371
ETTh1																				
96	*0.378*	**0.389**	0.382	0.400	0.394	0.408	0.392	0.405	0.386	0.392	0.429	0.440	0.396	0.440	0.427	0.450	**0.376**	0.419	0.449	0.459
192	**0.430**	**0.420**	0.440	0.427	0.446	0.438	0.441	0.436	0.440	0.430	0.494	0.482	0.434	0.460	0.472	0.486	0.420	0.448	0.500	0.482
336	*0.473*	0.442	**0.454**	**0.432**	0.485	0.455	0.501	0.478	0.480	*0.443*	0.706	0.689	0.462	0.476	0.527	0.527	0.459	0.465	0.521	0.496
720	**0.475**	*0.466*	*0.479*	**0.461**	0.495	0.474	0.538	0.526	0.486	0.472	0.750	0.689	0.494	0.500	0.644	0.605	0.506	0.507	0.514	0.512
Avg	**0.439**	**0.429**	**0.439**	*0.430*	0.455	0.444	0.468	0.461	0.448	0.434	0.595	0.575	*0.447*	0.469	0.518	0.517	0.440	0.460	0.496	0.487
ETTh2																				
96	*0.287*	*0.333*	**0.279**	**0.331**	0.294	0.343	0.331	0.381	0.290	0.339	0.632	0.547	0.364	0.407	0.304	0.359	0.346	0.388	0.358	0.397
192	*0.366*	*0.383*	**0.349**	**0.380**	0.378	0.394	0.432	0.435	0.379	0.395	0.876	0.663	0.466	0.458	0.394	0.422	0.429	0.439	0.456	0.452
336	0.413	0.420	**0.348**	**0.382**	*0.382*	*0.410*	0.441	0.451	0.421	0.431	0.924	0.702	0.479	0.476	0.385	0.421	0.496	0.487	0.482	0.486
720	0.417	0.435	**0.398**	**0.424**	*0.412*	*0.433*	0.564	0.578	0.436	0.453	1.390	0.863	0.487	0.492	0.463	0.475	0.463	0.474	0.515	0.511
Avg	0.372	*0.394*	**0.344**	**0.379**	*0.367*	0.395	0.442	0.461	0.382	0.405	0.956	0.694	0.449	0.458	0.387	0.419	0.434	0.447	0.453	0.462
Weather																				
96	*0.157*	*0.197*	**0.156**	**0.192**	0.177	0.218	0.195	0.253	0.168	0.208	0.181	0.231	0.288	0.365	0.202	0.261	0.238	0.314	0.249	0.329
192	*0.207*	*0.242*	**0.206**	**0.240**	0.224	0.258	0.239	0.299	0.217	0.255	0.219	0.275	0.368	0.425	0.242	0.298	0.275	0.329	0.325	0.370
336	*0.267*	*0.286*	**0.254**	**0.282**	0.277	0.297	0.282	0.333	0.267	0.292	0.274	0.332	0.447	0.469	0.287	0.335	0.339	0.377	0.351	0.391
720	*0.345*	*0.339*	**0.340**	**0.336**	0.350	0.345	0.352	0.390	0.351	0.346	0.356	0.387	0.640	0.574	0.351	0.386	0.389	0.409	0.415	0.426
Avg	*0.244*	*0.266*	**0.239**	**0.263**	0.257	0.280	0.267	0.319	0.251	0.275	0.258	0.306	0.436	0.458	0.271	0.320	0.310	0.357	0.335	0.379
Electricity																				
96	*0.146*	**0.233**	**0.145**	*0.236*	0.180	0.264	0.194	0.276	0.185	0.266	0.254	0.347	0.182	0.297	0.194	0.277	0.186	0.302	0.196	0.313
192	**0.163**	**0.247**	*0.167*	*0.256*	0.188	0.275	0.193	0.279	0.189	0.276	0.261	0.353	0.188	0.300	0.193	0.280	0.197	0.311	0.211	0.324
336	**0.178**	**0.263**	*0.186*	*0.275*	0.206	0.291	0.206	0.294	0.204	0.289	0.273	0.364	0.210	0.324	0.206	0.296	0.213	0.328	0.214	0.327
720	**0.213**	**0.293**	*0.231*	*0.309*	0.247	0.328	0.241	0.328	0.245	0.319	0.303	0.388	0.232	0.339	0.242	0.328	0.233	0.344	0.236	0.342
Avg	**0.175**	**0.259**	*0.182*	*0.269*	0.205	0.290	0.209	0.294	0.206	0.288	0.273	0.363	0.203	0.315	0.209	0.295	0.207	0.321	0.214	0.327
Traffic																				
96	**0.438**	**0.260**	*0.479*	*0.283*	0.492	0.324	0.648	0.396	0.645	0.388	0.558	0.320	2.678	1.071	0.568	0.352	0.576	0.359	0.597	0.371
192	**0.456**	**0.268**	*0.484*	*0.292*	0.487	0.303	0.613	0.386	0.599	0.365	0.572	0.331	0.564	0.351	0.612	0.371	0.610	0.380	0.607	0.382
336	**0.475**	**0.276**	*0.503*	*0.299*	0.505	0.317	0.614	0.383	0.606	0.367	0.587	0.342	0.570	0.349	0.605	0.374	0.608	0.375	0.623	0.387
512	**0.514**	**0.295**	*0.537*	*0.322*	0.542	0.337	0.655	0.405	0.645	0.388	0.652	0.359	0.576	0.349	0.647	0.410	0.621	0.375	0.639	0.395
Avg	**0.470**	**0.275**	*0.501*	*0.299*	0.507	0.320	0.632	0.393	0.624	0.377	0.592	0.338	1.097	0.530	0.608	0.377	0.604	0.372	0.617	0.384

Table 2. Multivariate time series forecasting results. The input length (lookback window) is fixed to 96 and prediction length is in {96, 192, 336, 720}. Bold and italics indicate the significant andsecond best values.

the Electricity and Traffic datasets, respectively. According to Table 3, our model consistently outperforms or achieves similar performance to LLM-based methods while using significantly fewer parameters. Notably, compared to Time-LLM, which utilizes 7 billion parameters, our model maintains competitive performance across different datasets. For example, when time series forecasting of the ETTh1 dataset at a prediction window of 720, MultiPatchFormer obtains an MSE of 0.434, lower than Time-LLM's 0.442 while consuming far fewer computational resources. This underscores the effectiveness of our approach relative to larger models. Graph learning models have been proved to be promising in forecasting traffic flow by modeling the temporal correlations and the spatial dependencies between the variables through the graph learning strategy[18,21]. We compare our MultiPatchFormer with the spatio-temporal graph models on various benchmarks, specially on Traffic dataset. As illustrated by Table 4, our model outperforms SDGL and MTGNN with a large margin, particularly on Traffic forecasting task, with average MSE improvement of 23% and 28%, respectively.

Methods	Ours		Time-LLM		GPT4TS	
Metric	MSE	MAE	MSE	MAE	MSE	MAE
Traffic						
96	0.370	**0.236**	**0.362**	0.248	0.388	0.282
192	0.383	**0.242**	**0.374**	0.247	0.407	0.290
336	0.398	**0.250**	**0.385**	0.271	0.412	0.294
720	0.433	**0.270**	**0.430**	0.288	0.450	0.312
Avg	0.396	**0.250**	**0.388**	0.264	0.414	0.294
Electricity						
96	**0.131**	**0.222**	0.131	0.224	0.139	0.238
192	**0.149**	**0.239**	0.152	0.241	0.153	0.251
336	0.162	0.253	**0.160**	**0.248**	0.169	0.266
720	**0.191**	**0.277**	0.192	0.298	0.206	0.297
Avg	**0.158**	**0.248**	**0.158**	0.252	0.167	0.263
ETTh1						
96	0.366	**0.392**	**0.362**	**0.392**	0.376	0.397
192	0.400	**0.418**	**0.398**	**0.418**	0.416	0.418
336	**0.423**	0.437	0.430	**0.427**	0.442	0.433
720	**0.434**	0.459	0.442	**0.457**	0.477	0.456
Avg	**0.406**	0.427	0.408	**0.423**	0.465	0.455
Weather						
96	**0.144**	**0.185**	**0.147**	0.201	0.162	0.212
192	0.190	**0.231**	**0.189**	0.234	0.204	0.248
336	**0.242**	**0.270**	**0.262**	0.279	0.254	0.286
720	0.313	0.323	**0.304**	**0.316**	0.326	0.337
Avg	**0.222**	**0.252**	0.225	0.257	0.237	0.270

Table 3. Comparison result of our model with LLM models. The input length (lookback window) is fixed to 512 and prediction length is in {96, 192, 336, 720}. Significant values are in bold.

As shown in Table 2, MultiPatchFormer consistently achieves low error rates for datasets with strong seasonality and highly correlated datasets with many variates (Electricity, Traffic, Weather). This can be attributed to the multi-scale embedding and temporal Transformer blocks, which effectively captures temporal correlations at various granularities and channel-wise attention, which represents the complex dependencies between time series channels through learning of the attention scores between channel pairs, extracting meaningful features. However, our model exhibits relatively lower performance on datasets with dominating short-term dependencies and limited training data, such as the ETTh2 dataset. This could be associated with the multi-scale embedding and multi-head attention, which, while being effective for capturing long-range patterns, may not show significant effectiveness for data with simpler temporal patterns and limited training samples.

In time series forecasting, size of the input series determines the historical information that a model receives to forecast. Different input lengths are configured to verify effectiveness of MultiPatchFormer to deal with various input lengths. As the input length increases, the error measures of MultiPatchFormer continue to decrease, demonstrating its robustness in handling long and noisy sequences evidenced by Table 5, which indicates an MSE reduction of more than 15%, 11% and 10% achieved on the ETTm1, Weather and Electricity datasets when prolonging the input length from 96 to 720. In this experiment, the best performing predictive models are selected and the results for input length of 48, 192 and 336 are visualized in Fig. 4. According to this figure, MultiPatchFormer consistently outperforms other baselines in most cases by using longer input lengths. This serves as evidence of efficiency of MultiPatchFormer in utilizing information of long input sequences thanks to multi-scale analysis and capturing cross-variable dependencies. MultiPatchFormer is also compared with other best performing baselines using a shorter input length (e.g. 48) on Electricity dataset. As Fig. 4 illustrates, our model forecasts different prediction lengths by using a short input window (H = 48) with the lowest error, indicating its capability in capturing temporal correlations from shorter input sequences.

Ablation studies

To verify the effectiveness of the components in our design of MultiPatchFormer, we conduct ablation studies by removing the multi-scale embedding, channel-wise encoder and multi-step decoder from the main model. Time series forecasting results using our model without those components are reported in Table 6. As evidenced by the table, each of the multi-scale embedding, channel-wise encoder and multi-step decoder modules contribute to performance promotion. For example, in ETTh1 forecasting dataset, multi-scale embedding improves the MSE error rate by approximately 2% in prediction length of 720 and the channel-wise encoder promotes the prediction accuracy (MSE) by 2.5%. Our multi-step decoder, improves the prediction error in most cases, specifically when the forecast horizon is long, e.g. 720. For example, in traffic forecasting, consisting of 862

Methods	Ours		SDGL		MTGNN	
Metric	MSE	MAE	MSE	MAE	MSE	MAE
Traffic						
96	**0.438**	**0.260**	0.557	0.271	0.660	0.437
192	**0.456**	**0.268**	0.581	0.287	0.649	0.438
336	**0.475**	**0.276**	0.611	0.302	0.653	0.472
720	**0.514**	**0.295**	0.702	0.345	0.639	0.437
Avg	**0.470**	**0.275**	0.613	0.301	0.650	0.446
Electricity						
96	0.146	**0.233**	**0.145**	0.238	0.217	0.318
192	0.163	**0.247**	**0.159**	0.255	0.238	0.352
336	**0.178**	**0.263**	0.185	0.284	0.260	0.348
720	**0.213**	**0.293**	0.232	0.314	0.290	0.369
Avg	**0.175**	**0.259**	0.180	0.273	0.251	0.347
ETTh1						
96	**0.378**	**0.389**	0.446	0.428	0.515	0.517
192	**0.430**	**0.420**	0.471	0.452	0.553	0.522
336	**0.473**	**0.442**	0.506	0.475	0.612	0.577
720	**0.475**	**0.466**	0.623	0.548	0.609	0.597
Avg	**0.439**	**0.429**	0.512	0.476	0.572	0.553
Weather						
96	0.157	0.197	**0.153**	**0.194**	0.230	0.329
192	**0.207**	**0.242**	0.212	0.257	0.263	0.322
336	0.267	**0.286**	**0.263**	0.294	0.354	0.396
720	**0.345**	**0.339**	**0.352**	0.360	0.409	0.371
Avg	**0.244**	**0.266**	0.245	0.276	0.314	0.355

Table 4. Comparison result of our model with graph models. The input length (lookback window) is set to 96 and prediction length is in {96, 192, 336, 720}. Significant values are in bold.

Input length	96		192		336		512		720	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	0.396	0.389	0.365	0.379	0.352	0.372	**0.350**	**0.372**	0.353	0.376
ETTm2	0.279	0.321	0.263	0.315	0.253	**0.307**	**0.252**	0.309	0.256	0.313
Weather	0.249	0.270	0.232	0.259	0.224	0.254	**0.222**	**0.254**	0.224	0.257
ECL	0.175	0.259	0.163	0.250	0.161	0.249	0.159	0.249	**0.158**	**0.248**

Table 5. Multivariate time series forecasting results of MultiPatchFormer (ours) with different input length {96, 192, 336, 512, 720} averaged over four prediction lengths. Significant values are in bold.

variables across 720 future timestamps, the utilization of a multi-step decoder yields an MAE error reduction of 1%. We utilize a different kernel size for each dataset in the channel summarization part of the channel-wise attention in order to project the key and values, depending on the performance improvement. In some cases, e.g., Electricity dataset, the kernel size is set to 1, since it gives the best results compared to the larger kernels. But, in datasets with highly correlated channels (such as Traffic with 862 variates), large kernels (e.g., 21) yield lower error rates by reducing channel dimension in key and value of the channel-wise attention. We study the impact of varying number of scales on time series forecasting and indicate the results in Table 7. In some cases, considering only one dominant scale is enough, but in some datasets, utilizing two or more than two scales helps to boost the performance and capture cross-scale information. In addition, more experiments are conducted in order to verify the effect of hyper-parameters, including the learning rate, hidden (model) dimension and number of training epoch. The results are illustrated in Fig. 5.

Model efficiency

Our model is versatile in terms of speed and memory usage efficiency. Specifically, MultiPatchFormer is comparable with other most recent models, including PatchTST[7] and Pathformer[8] in terms of training time and memory footprint. As illustrated in Fig. 6, our model is faster than its main competitor model (Pathformer) in terms of training time, and its memory footprint is lower than most of the baselines, being close to memory usage of the linear models (DLinear). In time-series forecasting of ETTh1 dataset, with input length of 96 and prediction length of 96, our model is more than ten times faster than Pathformer and FEDformer, demonstrating

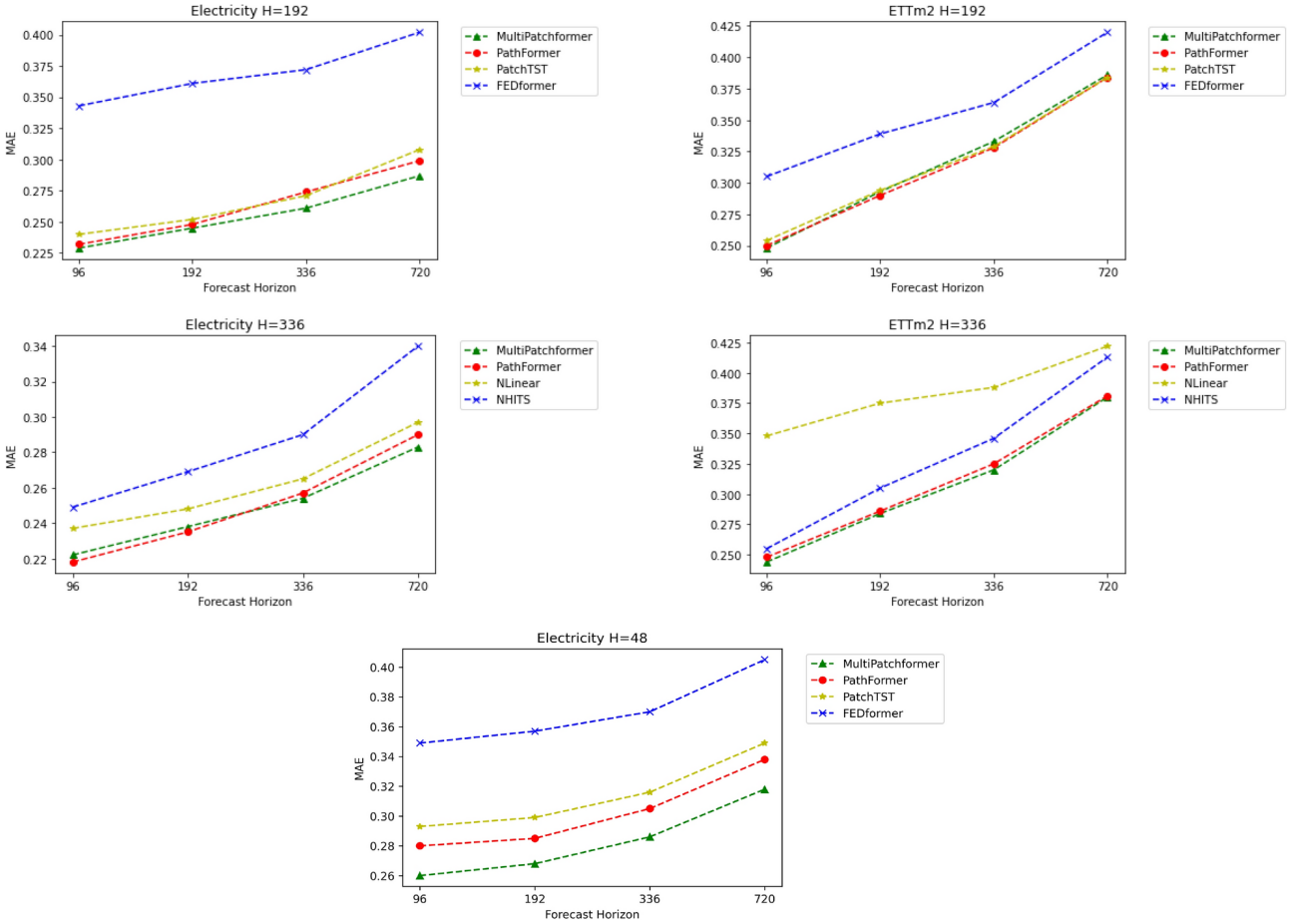


Fig. 4. Time series forecasting results of MultiPatchFormer with different input lengths (H = 48, 192 and 336) on ETTm2 and Electricity datasets.

	W/O multi-scale embedding		W/O channel-wise encoder		W/O multi-step decoder		MultiPatchFormer	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1								
96	0.383	0.394	0.387	**0.393**	0.382	0.395	**0.380**	**0.393**
192	0.435	**0.424**	0.443	0.425	0.435	0.427	**0.432**	**0.424**
336	0.475	**0.444**	0.485	0.445	0.474	0.449	**0.472**	**0.444**
720	0.478	0.470	0.481	**0.464**	0.473	0.471	**0.469**	**0.464**
Traffic								
96	**0.437**	0.264	0.490	0.311	0.439	**0.260**	0.439	**0.260**
192	0.460	0.271	0.493	0.306	**0.456**	**0.269**	0.457	**0.269**
336	0.483	0.280	0.503	0.301	**0.477**	0.278	0.478	**0.277**
720	0.516	0.300	0.537	0.319	0.516	0.299	**0.515**	**0.297**
Electricity								
96	0.147	0.235	0.167	0.248	**0.146**	0.234	**0.146**	**0.233**
192	0.164	0.248	0.181	0.259	**0.163**	**0.247**	**0.163**	**0.247**
336	0.179	0.265	0.196	0.275	**0.178**	0.264	**0.178**	**0.263**
720	0.216	0.295	0.235	0.308	0.216	0.295	**0.213**	**0.293**

Table 6. Ablation of MultiPatchFormer model without different components: multi-scale embedding, channel-wise attention and multi-step decoder. Significant values are in bold.

Number of scales	1		2		4	
Dataset	MAE	MSE	MAE	MSE	MAE	MSE
Electricity						
96	0.138	0.229	**0.128**	**0.218**	0.131	0.222
192	0.150	**0.239**	0.155	0.244	**0.149**	**0.239**
336	0.165	0.255	0.168	0.258	**0.162**	**0.253**
720	0.195	0.280	0.192	**0.276**	**0.191**	0.277
ETTm1						
96	**0.309**	**0.338**	0.310	0.339	0.313	0.343
192	**0.362**	**0.369**	0.368	0.370	0.365	**0.369**
336	0.398	**0.393**	**0.396**	**0.393**	**0.396**	**0.393**
720	0.472	0.434	**0.465**	**0.433**	0.470	0.434
ETTh1						
96	0.380	0.390	0.379	**0.389**	**0.378**	**0.389**
192	0.432	**0.420**	0.432	0.421	**0.430**	**0.420**
336	0.477	0.444	0.474	0.442	**0.473**	**0.441**
720	0.482	0.471	0.486	0.476	**0.476**	**0.468**

Table 7. Ablation study on the number of scales in multi-scale embedding. Significant values are in bold.

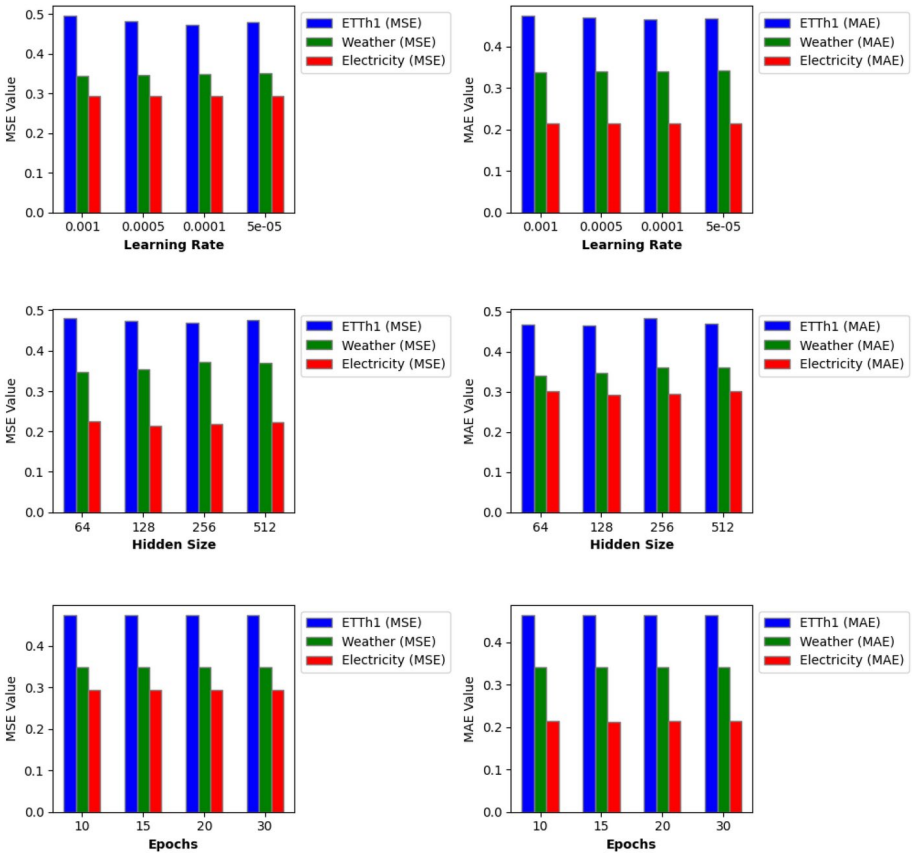


Fig. 5. Influence of hyperparameters on time series forecasting of MultiPatchFormer.

lower memory consumption, while delivering the similar or better prediction accuracy. This highlights MultiPatchFormer's efficiency while providing accurate forecasts.

Visualization

We visualize prediction results of MultiPatchFormer for Electricity and Traffic datasets. As illustrated in Figs. 7 and 8, the prediction curves align well with the ground truth ones in various cases and prediction horizons, indicating MultiPatchFormer's capability in handling complex trends and patterns.

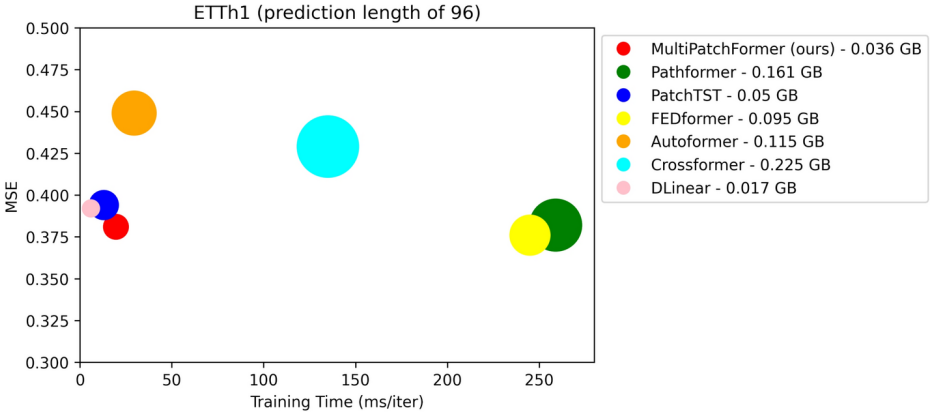


Fig. 6. Model efficiency comparison. Greater circle diameter represents larger memory footprint (in GB).

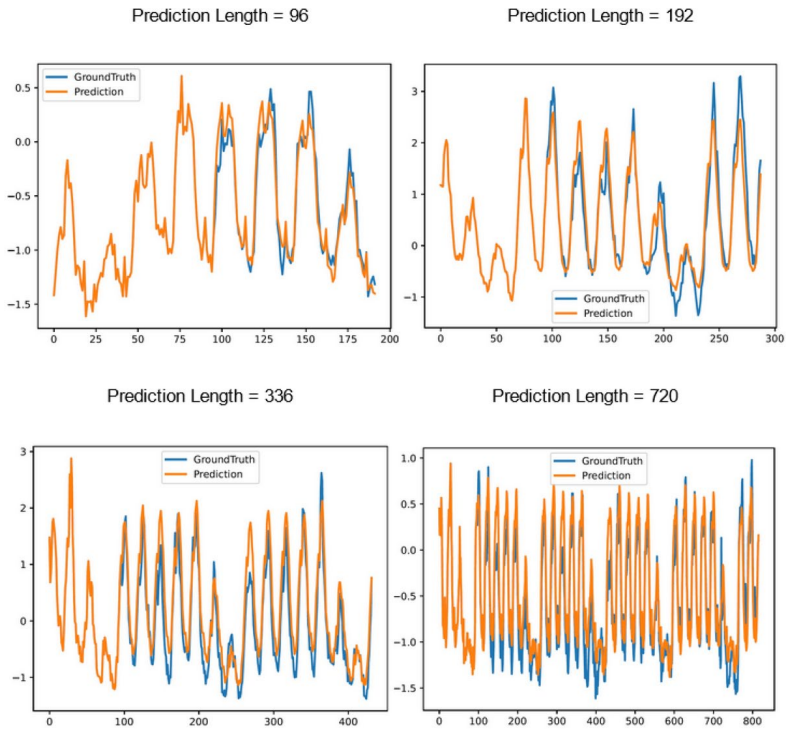


Fig. 7. Prediction results of our model applied to the Electricity dataset with different forecasting horizons.

Conclusion

We propose MultiPatchFormer, a Transformer based time series forecasting model, which integrates temporal dependencies associated with different temporal scales and captures intricate correlations among time series channels. In addition, we utilize 1-dimensional convolution to reduce channel dimension of key and value in the channel-wise multi-head attention to decrease noise effect and computational burden. A novel multi-step decoder is further devised to reduce over-fitting effect in dealing with long forecasting horizons. These innovative components collectively empower our model to produce accurate forecasts in variety of domains. Extensive experiments show the superior performance and robustness of our model in different settings and with varying input lengths.

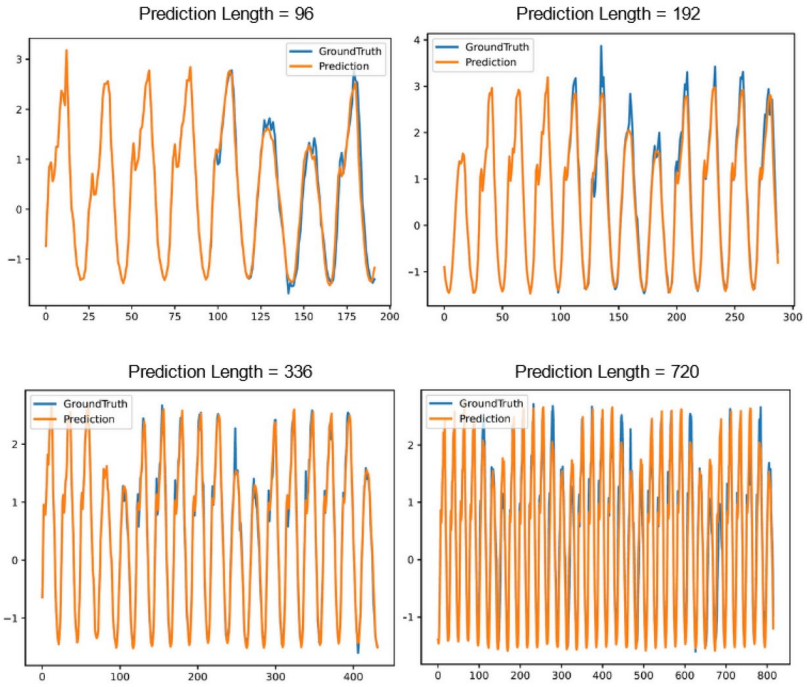


Fig. 8. Prediction results of our model applied to the Traffic dataset with different forecasting horizons.

Data availability

All the datasets used in this work are publicly available[24]. The link of the aforementioned repository to download the datasets is: https://drive.google.com/drive/folders/1ZOYpTUa82_jCcxIdTmyr0LXQfvaM9vIy.

Code availability

Accession codes The code is available at: https://anonymous.4open.science/r/MultiPatchFormer-DD43.

Received: 6 September 2024; Accepted: 5 December 2024
Published online: 10 January 2025

References

1. Brown, T. et al. Language models are few-shot learners. *Adv. Neural Inform. Process. Syst.* **33**, 1877–1901 (2020).
2. Dosovitskiy, A. An image is worth 16×16 words: Transformers for image recognition at scale. *arXiv preprint*. arXiv:2010.11929 (2020).
3. Vaswani, A. *et al.* Attention is all you need. *Adv. Neural Inform. Process. Syst.* **30** (2017).
4. Zhou, T. *et al.* Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. in *International conference on machine learning*, 27268–27286 (PMLR, 2022).
5. Zhou, H. et al. Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proc. AAAI Conf. Artif. Intell.* **35**, 11106–11115 (2021).
6. Zeng, A., Chen, M., Zhang, L. & Xu, Q. Are transformers effective for time series forecasting?. *Proc. AAAI Conf. Artif. Intell.* **37**, 11121–11128 (2023).
7. Nie, Y., Nguyen, N. H., Sinthong, P. & Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint*. arXiv:2211.14730 (2022).
8. Chen, P. *et al.* Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting. *arXiv e-prints* arXiv-2402 (2024).
9. Ferreira, M. A., Higdon, D. M., Lee, H. K. & West, M. Multi-scale and hidden resolution time series models. *Bayesian Anal.*, 947–967 (2006).
10. Liu, Y. *et al.* itransformer: Inverted transformers are effective for time series forecasting. in *The Twelfth International Conference on Learning Representations* (2023).
11. Shabani, A., Abdi, A., Meng, L. & Sylvain, T. Scaleformer: Iterative multi-scale refining transformers for time series forecasting. *arXiv preprint*. arXiv:2206.04038 (2022).
12. Elsaraiti, M., Ali, G., Musbah, H., Merabet, A. & Little, T. Time series analysis of electricity consumption forecasting using arima model. in *2021 IEEE Green technologies conference (GreenTech)*, 259–262 (IEEE, 2021).
13. Hyndman, R. J. & Khandakar, Y. Automatic time series forecasting: The forecast package for r. *J. Stat. Softw.* **27**, 1–22 (2008).
14. Salinas, D., Flunkert, V., Gasthaus, J. & Januschowski, T. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *Int. J. Forecasting* **36**, 1181–1191 (2020).
15. Wu, H. *et al.* Timesnet: Temporal 2d-variation modeling for general time series analysis. in *The Eleventh International Conference on Learning Representations*. (2022).
16. Liu, M. et al. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Adv. Neural Inform. Process. Syst.* **35**, 5816–5828 (2022).
17. Das, A. *et al.* Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint*. arXiv:2304.08424 (2023).
18. Wu, Z. *et al.* Connecting the dots: Multivariate time series forecasting with graph neural networks. in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 753–763 (2020).

19. Guo, S., Lin, Y., Wan, H., Li, X. & Cong, G. Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting. *IEEE Trans. Knowl. Data Eng.* **34**, 5415–5428 (2021).
20. Wang, S., Zhang, M., Miao, H., Peng, Z. & Yu, P. S. Multivariate correlation-aware spatio-temporal graph convolutional networks for multi-scale traffic prediction. *ACM Trans. Intell. Syst. Technol. (TIST)* **13**, 1–22 (2022).
21. Li, Z. L., Zhang, G. W., Yu, J. & Xu, L. Y. Dynamic graph structure learning for multivariate time series forecasting. *Pattern Recognit.* **138**, 109423 (2023).
22. Zhao, K. et al. Multiple time series forecasting with dynamic graph modeling. *Proc. VLDB Endowment* **17**, 753–765 (2023).
23. Miao, H. *et al.* Less is more: Efficient time series dataset condensation via two-fold modal matching–extended version. *arXiv preprint*. arXiv:2410.20905 (2024).
24. Wu, H., Xu, J., Wang, J. & Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Adv. Neural Inform. Process. Syst.* **34**, 22419–22430 (2021).
25. Cirstea, R.-G. *et al.* Triformer: Triangular, variable-specific attentions for long sequence multivariate time series forecasting–full version. *arXiv preprint*. arXiv:2204.13767 (2022).
26. Jin, M. *et al.* Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint*. arXiv:2310.01728 (2023).
27. Zhou, T. et al. One fits all: Power general time series analysis by pretrained lm. *Adv. Neural Inform. Process. Syst.* **36**, 43322–43355 (2023).
28. Radford, A. et al. Language models are unsupervised multitask learners. *OpenAI Blog* **1**, 9 (2019).
29. Liu, C. *et al.* Spatial-temporal large language model for traffic prediction. *arXiv preprint*. arXiv:2401.10134 (2024).
30. Wang, S., Zhang, M., Miao, H. & Yu, P. S. Mt-stnets: Multi-task spatial-temporal networks for multi-scale traffic prediction. in *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, 504–512 (SIAM, 2021).
31. Kim, T. *et al.* Reversible instance normalization for accurate time-series forecasting against distribution shift. in *International Conference on Learning Representations* (2021).
32. Zhang, Y. & Yan, J. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. in *The Eleventh International Conference on Learning Representations* (2022).
33. Liu, S. *et al.* Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. in *International Conference on Learning Representations* (2021).
34. Paszke, A. *et al.* Pytorch: An imperative style, high-performance deep learning library. *Adv. Neural Inform. Process. Syst.* **32** (2019).

Author contributions

V.N. conceived the experiment(s), V.N. and A.D. conducted the experiment(s), M.B. and A.D. analysed the results. All authors reviewed the manuscript.

Declaration

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to A.B.D.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

AI

MDPI

Article

Should We Reconsider RNNs for Time-Series Forecasting?

Vahid Naghashi [1,2], Mounir Boukadoum [1,2] and Abdoulaye Banire Diallo [1,2,*]

[1] Department of Computer Science, Université du Québec à Montréal, Montreal, QC H2L 2C4, Canada; naghashi.vahid@courrier.uqam.ca (V.N.); boukadoum.mounir@uqam.ca (M.B.)
[2] WELL-E : Research and Innovation Chair in Animal Welfare and Artificial Intelligence, Montreal, QC H2L 2C4, Canada
* Correspondence: diallo.abdoulaye@uqam.ca

Abstract: (1) Background: In recent years, Transformer-based models have dominated the time-series forecasting domain, overshadowing recurrent neural networks (RNNs) such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). While Transformers demonstrate superior performance, their high computational cost limits their practical application in resource-constrained settings. (2) Methods: In this paper, we reconsider RNNs—specifically the GRU architecture—as an efficient alternative to time-series forecasting by leveraging this architecture's sequential representation capability to capture cross-channel dependencies effectively. Our model also utilizes a feed-forward layer right after the GRU module to represent temporal dependencies, and aggregates it with the GRU layers to predict future values of a given time-series. (3) Results and conclusions: Our extensive experiments conducted on different real-world datasets show that our inverted GRU (iGRU) model achieves promising results in terms of error metrics and memory efficiency, challenging or surpassing state-of-the-art models on various benchmarks.

Keywords: time-series; gated recurrent units; temporal dependencies; cross-channel correlations

Academic Editor: Mohammad Valipour

Received: 2 April 2025
Revised: 23 April 2025
Accepted: 24 April 2025
Published: 25 April 2025

Citation: Naghashi, V.; Boukadoum, M.; Diallo, A.B. Should We Reconsider RNNs for Time-Series Forecasting? *AI* **2025**, *6*, 90. https://doi.org/10.3390/ai6050090

1. Introduction

Time-series forecasting is an important task in several application domains, including finance, meteorology, transportation, and energy consumption. Providing accurate forecasts helps industries and businesses save both money and time by enabling optimized and informed decision-making ahead of time, thereby avoiding unnecessary actions. Time-series forecasting involves leveraging historical (past) data from various channels or variates to predict future values of the same or related variates. As shown in Figure 1, these variables are often inter-correlated, and temporal relationships exist along the time dimension in a time-series. Time-series analysis and forecasting have garnered significant attention from researchers over the past few decades. With the rise of Artificial Intelligence (AI), deep learning-based methods have taken the lead in this field [1]. Among the deep learning models developed for time-series forecasting, Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), Multi-Layer Perceptrons (MLPs), Transformer models, and Large Language Model (LLM)-based approaches have attained remarkable performance due to their ability to capture complex long-term temporal dependencies [1–5]. A model usually demonstrates high performance for multivariate time-series forecasting when it captures the relations between the prediction variables and the temporal correlations across the historical time steps. There are two main approaches used for time-series forecasting with deep learning models. The first category of methods are known as Channel-Dependent (CD) methods, which usually project the channel dimension into a hidden space

(modeling dimension). However, recent works have shown that Channel-Independent (CI) models generally achieve better results [6,7]. Recently, many Transformer-based models have been proposed using channel independence, where multiple channels are predicted independently. In addition, one of the recently invented CI-based models, iTransformer [8], directly captures the intricate interaction among multiple time-series channels by exploiting the high capacity of the multi-head self-attention module inside the Transformer, and embedding the temporal dimension within the model dimension, leading to impressive results, specifically when dealing with complex real-world datasets. However, the Transformer-based models face the obvious challenge of quadratic time complexity with respect to the input sequence length, specifically in the inference step, which gives rise to an intensive calculation when applied to a large number of variates or longer look-back windows, hindering their deployment in real-world applications.

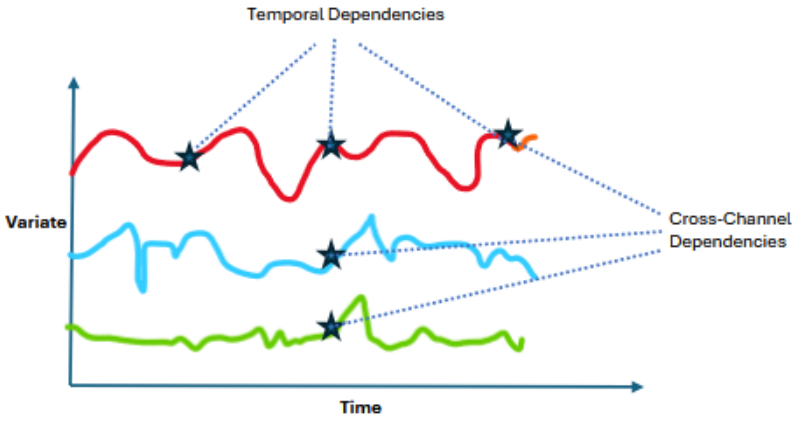


Figure 1. An example of a multivariate time-series which consists of multiple channels (variates). There are inter-variate dependencies between the channels and temporal correlations between different time-steps.

There have already been attempts to reduce the computational complexity of Transformer for time-series forecasting. For instance, Ref. [9] modified the Transformer to focus on a portion of the sequence, while other works utilized linear models to decrease the time complexity [7,10]. Although linear models can reduce time complexity, they mainly rely on linear computations and fail to exploit contextual information, resulting in sub-optimal predictions, specifically when applied to datasets involving many variates or series. Given the prevalent use of Transformer models in the time-series forecasting (TSF) domain, recurrent models such as LSTM and GRU have been neglected and their potential to capture cross-channel dependencies escaped the attention of researchers in the field. RNNs are still occasionally utilized for TSF [11] in rare cases. However, their capability for sequential modeling is underestimated compared to that of Transformers.

In this paper, we reconsider the RNNs, specifically the Gated Recurrent Unit (GRU), as an alternative to the multi-head self-attention module in the Transformer models. We exploit GRUs to extract the interactions among time-series channels and utilize them in multiple-channel forecasting. Inspired by the iTransformer [8] model, we apply the GRU to the channel dimension of the input series to capture inter-series correlations. Considering the importance of capturing cross-channel correlations in time-series analysis, we will answer the question of whether RNNs (including GRU or LSTM) should still be considered for time-series forecasting. We answer this question by conducting experiments using various public time-series datasets and analyzing the results, specifically through the ablation experiments. Recurrent Neural Networks (RNNs) represent significant advantages in inference time efficiency for time-series forecasting, although their performance some-

times falls behind that of Transformer-based architectures. The proposed iGRU achieves a competitive performance with reduced computational overhead. On datasets like Solar, iGRU outperforms several Transformer models in both accuracy and efficiency, as shown in Section 3. However, for some datasets, iGRU's performance is surpassed by computation-intensive models, reflecting a trade-off between accuracy and computational cost. This work explores these trade-offs, demonstrating iGRU's potential as a lightweight and effective model for time-series forecasting task. Furthermore, in most cases, iGRU outperforms the recently proposed iTransformer [8]. The prominent reason for utilizing GRUs in our model is that RNNs provide significantly lower time and memory complexity compared to many Transformer-based models, resulting in improved efficiency for specific applications, as demonstrated in Table 1. Additionally, the clear temporal flow in RNNs provides better interpretability in understanding how information propagates through sequences [12], specifically when compared to Transformer and MLP-based architectures.

Table 1. Comparison of time and memory complexity for different models, where L represents the input sequence length (context length).

Method	Type	Time Complexity	Memory Complexity
GRU	RNN	$O(L)$	$O(L)$
DLinear	MLP	$O(L)$	$O(L)$
Crossformer	Transformer	$O(L^2)$	$O(L^2)$

Our work includes the following contributions to the time-series forecasting domain:

- We reconsider RNNs for time-series forecasting using a different approach by focusing on the inter-channel dependencies and describe the inverted GRU (iGRU), which exploits GRU blocks to capture interactions between the time-series channels and feed-forward layers to represent temporal relations.
- We extensively evaluate iGRU on eleven public datasets and report the results in terms of error metrics and memory efficiency.
- We show that our iGRU model achieves comparable results to the state-of-the-art models or outperforms them.

Time-series forecasting methods are generally categorized into statistical models, such as Auto-ARIMA [13], and modern (deep learning) approaches, including Transformer, linear and convolutional neural network models. The Transformer architecture was initially designed to process and generate token sequences, mainly for natural language processing applications, especially Large Language Models (LLMs). However, its excellent potential motivated the TSF research community to deploy and adapt it for time-series tasks. For instance, LogTrans [14] uses convolutional attention in the LogSparse design to capture local information and reduce time complexity. The Informer [15] exploits the ProbSparse self-attention with distillation to emphasize prominent keys. In the Autoformer [16], the idea of time-series decomposition and auto-correlation calculation is proposed to extract temporal correlations. The FEDformer [17] is designed based on Fourier-based architecture and achieves a linear time complexity. In another work, the Pyraformer [18] utilizes pyramidal attention to capture inter-scale and intra-scale relations with a linear complexity. Recently, PatchTST [19] was proposed based on a Transformer architecture, which utilizes patched time-series with channel independence to capture temporal correlations for each channel, separately. In a different design, the Crossformer [20] exploits an encoder–decoder structure with hierarchical attention modules to leverage cross-channel dependencies. However, some Linear models have emerged recently to outperform Transformers in benchmark experiments [7,21]. On the other hand, these linear models fall short in representing non-linear

dependencies between the input series and future time steps [7]. Recently, CNN-based models achieved promising results in time-series analysis tasks. As a prominent example, TimesNet [22] utilizes two-dimensional convolutions to capture inter-period and intra-period relations in a time-series with multiple period lengths, obtaining promising results. Over the past few years, Transformers and CNNs have overshadowed Recurrent Neural Networks (RNNs) in the time-series forecasting domain. For instance, the iTransformer [8] model is proposed based on the vanilla Transformer model and embedding the channels of the input series. This model attained impressive results in many benchmark datasets, pinpointing the importance of modeling cross-channel interactions in the forecasting tasks. The more recent model, TimeXer [23], incorporates the external information to enhance the forecasting accuracy, which strengthens the canonical Transformer to harmonize endogenous and exogenous information by using patch-wise self-attention and cross-variate attention, simultaneously. In this paper, we reconsider the potential of RNNs for capturing sequential dependencies and introduce the inverted GRU (iGRU), which leverages GRUs to capture dependencies across time-series channels while utilizing feed-forward layers to extract temporal features.

2. Materials and Methods

Our iGRU architecture is illustrated in Figure 2 and the corresponding forecasting procedure is shown in Algorithm 1. Given $x_t \in \mathbb{R}^C$, the observation of a time-series with C channels at time t, we aim to forecast its future H time-steps $x_{t+1}, ..., x_{t+H}$ using a history or context window w_t of length L (i.e., $w_t = (x_{t-L+1}, ..., x_t)$).

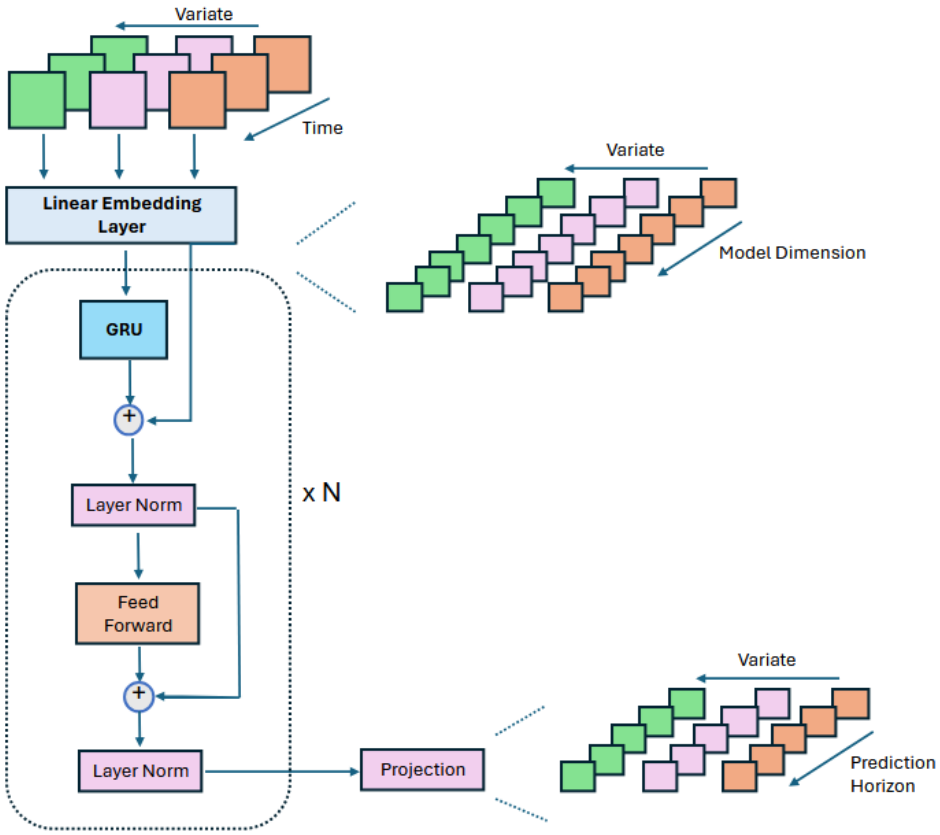


Figure 2. Illustration of iGRU architecture.

Algorithm 1 The forecasting procedure of iGRU

Input: Batch$(X) = [x_1, x_2, \ldots, x_L] : (B, L, C)$
Output: Batch$(Y) = [y_1, y_2, \ldots, y_H] : (B, H, C)$

$X_T : (B, C, L) \leftarrow$ Transpose(Batch(X))
$X_{embedded} : (B, C, D) \leftarrow$ Embedding(X_T)
for each layer l in iGRU layers **do**
  $X_{CC} \leftarrow \overrightarrow{\text{GRU}}(X_{embedded})$
  $X_{CCR} \leftarrow X_{CC} + X_{embedded}$
  $X_{CCR} \leftarrow$ LayerNorm(X_{CCR})
  $X_{FF} \leftarrow$ Feed-Forward(X_{CCR})
  $X_{FF} \leftarrow X_{FF} + X_{CCR}$
  $X_F \leftarrow$ LayerNorm(X_{FF})
  $X_{embedded} \leftarrow X_F$
end for
$X_{out} : (B, C, H) \leftarrow$ Projection(X_F)
Batch$(Y) : (B, H, C) \leftarrow$ Transpose(X_{out})

2.1. Preliminaries

2.1.1. RNN and GRU

Recurrent Neural Networks (RNNs) are a category of neural networks designed for sequential data processing across multiple time steps. The Gated Recurring Unit [24], introduced in 2014, is a specific type of RNN with a gating mechanism to input or forget information along a sequence of timesteps, which employs update and forget units to process sequential data mapped to a hidden space. The GRU operation is defined by the following equations:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{1}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{2}$$

$$\hat{h}_t = \phi(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{3}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \tag{4}$$

where $\odot$ represents element-wise multiplication. x_t and h_t indicate input and hidden state vectors at time t, respectively. z_t, r_t, $\hat{h}_t$ represent the update vector, reset vector and candidate hidden state at time step t, respectively. In addition, W_h, W_z, W_r, U_h, U_z, U_r, b_h, b_z and b_r are the relevant weights and biases which are learned during the model training phase. This type of RNN showed effective sequential modeling in various applications [24–26]. In most applications, GRUs and LSTMs are commonly used to capture temporal dependencies. However, their efficiency in modeling cross-channel correlations within time-series data has often been overlooked. To leverage RNNs for capturing inter-series dependencies, the input multivariate time-series must first be projected into a hidden space using a simple linear embedding layer [8]. We selected GRU over LSTM due to its comparatively lower parameter count while achieving similar performance, which aligned with our goal of maintaining model efficiency. Vanilla RNNs, on the other hand, are not studied in our approach as they lack memory gates, which limits their performance relative to GRUs.

2.1.2. Temporal Embedding of Time-Series

Similarly to [8], the input multivariate series is embedded into a higher-dimensional space through a linear layer. The input to the temporal embedding has the shape X (Batch, Channel, Time Length), which is an inverted version of the input time-series obtained by

swapping the temporal and channel dimensions. Then, X is projected into the model space along its temporal dimension by

$$X_{embedded} = Linear(X) \tag{5}$$

Here, *Linear* refers to a fully connected layer, where the input dimension corresponds to the series length, and the output dimension corresponds to the model dimensionality.

2.2. Proposed iGRU Model

As illustrated in Figure 2, the input multivariate series is first passed to the linear embedding layer, which is applied series-wise to map each channel into a hidden space. Before embedding, instance normalization [27] is utilized to reduce non-stationarity and distribution shifts between the training and test sets. The embedded series is then sent to the GRU module, which acts like the multi-head self-attention of Transformers to capture the intricate interactions among multiple time-series channels. Here, the GRU cells represent those dependencies in one direction starting from the first channel to the last one, similar to temporal sequence modeling:

$$X_{CC} = \overrightarrow{GRU}(X_{embedded}) \tag{6}$$

The channel or variate correlated output, X_{CC}, encoded by the GRU layer, is connected with its input to form the output of this layer, facilitating gradient flow and training stability:

$$X_{CCR} = X_{CC} + X_{embedded} \tag{7}$$

After adding the GRU output and the skip connection, layer normalization is applied to normalize the activations within each layer to obtain a mean of zero and a variance of one, thereby stabilizing training through the reduction of varying feature scales. Then, a feed-forward layer (FFN) is applied to the series representations to capture temporal correlations along each channel. The feed-forward network (FFN) consists of two fully connected layers mapping the series representation to a higher dimensional space (two times the model dimension) and then to the model dimension, with a Gaussian Error Linear Unit (GELU) activation used between the layers. It is worth noting that FFN implicitly represents temporal dependencies along the model dimension. Finally, the FFN module output is added to its input using a skip connection and a normalization layer is employed afterwards to adjust the obtained series representation. The final prediction is obtained by applying a projection layer to the output of the feed-forward network. The projection layer is a fully connected layer which projects the model dimension to the forecasting horizon (prediction length), generating the predictions of multiple channels.

3. Results

The Proposed iGRU is thoroughly and carefully evaluated on eleven public datasets. The Traffic dataset [6] is a collection of road occupancy data from the California Department of Transportation. It was gathered from 862 sensors between 2015 and 2016. PEMS [6,28] is a complex spatiotemporal dataset related to public traffic networks in California consisting of four different datasets (PEMS03, PEMS04, PEMS07, and PEMS08). The ETT (Electricity Transformer Temperature) dataset [16] includes data related to the load and oil temperature of electricity transformers collected from July 2016 to July 2018. This collection contains different datasets captured hourly or in minutes granularity, including ETTm1 and ETTm2, all consisting of seven variates. The Weather dataset [16] consists of 21 meteorological variates recorded every 10 min from the Max Planck Bio-geochemistry institute. The Electricity dataset [16] contains hourly electricity consumption of 321 costumers. The Solar-

Energy dataset [16], which was sampled every 10 min, collected solar power records in 2006 from 137 PV plants in the US state of Alabama. The Exchange dataset [16] is collected based on panel data of daily exchange rates corresponding to eight countries from 1990 to 2016. More information regarding the datasets are reported in Table 2.

Table 2. Dataset information: Dim represents the number of variates and Dataset Size denotes the total number of time points in (Training, Validation, Testing) split of each dataset, respectively. Prediction Length indicates the future time points to be predicted and four prediction lengths settings are specified in each dataset. Frequency denotes the sampling interval of time points.

Dataset	Dim	Prediction Length	Dataset Size	Frequency	Information
ETTm1, ETTm2	7	{96, 192, 336, 720}	(34,465, 11,521, 11,521)	15 min	Electricity
Exchange	8	{96, 192, 336, 720}	(5120, 665, 1422)	Daily	Economy
Weather	21	{96, 192, 336, 720}	(36,792, 5271, 10,540)	10 min	Weather
Electricity	321	{96, 192, 336, 720}	(18,317, 2633, 5261)	Hourly	Electricity
Traffic	862	{96, 192, 336, 720}	(12,185, 1757, 3509)	Hourly	Transportation
Solar-Energy	137	{96, 192, 336, 720}	(36,601, 5161, 10,417)	10 min	Energy
PEMS03	358	{12, 24, 48, 96}	(15,617, 5135, 5135)	5 min	Transportation
PEMS04	307	{12, 24, 48, 96}	(10,172, 3375, 3375)	5 min	Transportation
PEMS07	883	{12, 24, 48, 96}	(16,911, 5622, 5622)	5 min	Transportation
PEMS08	170	{12, 24, 48, 96}	(10,690, 3548, 3548)	5 min	Transportation

We compared the proposed iGRU model with several state-of-the-art models, including TimeXer [23], iTransformer [8], PatchTST [19], Crossformer [20], TiDE [21], DLinear [7], FEDformer [17], Autoformer [16] and TimesNet [22]. We implemented our proposed model in Pytorch [29] and executed our experiments using a single A100-40G NVIDIA GPU. All models were trained for 10 epochs with early stopping patience of three steps based on the validation loss change. The Mean Square Error (MSE) loss function is utilized with the Adam [30] optimizer to train all models. We set the learning rate to 0.001 for the Traffic, Electricity and PEMS datasets and lower values (0.0005 or 0.0001) are used for the other datasets. This choice is an attempt to mitigate overfitting and skipping of sub-optimal results, since these datasets had a limited number of training instances. For each hyper-parameter, we tried a range of possible values, and the value yielding the best result was picked. The reported results represent an average of five runs. In our experiments, the batch size is uniformly selected as 16 or 32, and the number of iGRU blocks are set from $\{1, 2, 3, 4\}$. Additionally, the model dimension is chosen from $\{128, 256, 512\}$ according to each dataset. The selected hyper-parameters for each dataset are shown in Table 3. The results of time-series forecasting in terms of MSE and MAE (Mean Absolute Error) are reported in Tables 4 and 5.

Table 3. Selected hyper-parameters for training the iGRU model on different benchmarks.

Dataset	Model Dimension	Feed-Forward Dimension	iGRU Blocks	Learning Rate	Batch Size	Dropout
ETTm1	256	512	2	0.0001	32	0.1
ETTm2	256	512	2	0.0001	32	0.1
Weather	512	512	3	0.0001	32	0.1
Exchange	256	256	2	0.00005	32	0.1
Electricity	512	512	3	0.001	16	0.1
Traffic	512	512	4	0.001	16	0.1
PEMS03	512	512	4	0.001	16	0.1
PEMS04	1024	1024	4	0.001	16	0.1
PEMS07	512	512	3 or 4	0.001	16	0.1
PEMS08	512	512	3 or 4	0.001	16	0.1

Table 4. Multivariate time-series forecasting results of iGRU and the baseline models on Traffic and PEMS datasets. The input length (lookback window) is set to 96 and the prediction length is in {12, 24, 48, 96} for PEMS datasets and in {96, 192, 336, 720} for Traffic dataset. The best results are shown in bold, and the second-best results are shown in italics.

Models		**Ours**		**TimeXer**		**iTransformer**		**PatchTST**		**DLinear**		**Crossformer**		**TimesNet**		**TiDE**		**FEDformer**		**Autoformer**	
	Metric	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
Traffic	96	**0.393**	**0.268**	0.428	0.271	0.395	0.268	0.462	0.295	0.650	0.396	0.522	0.290	0.593	0.321	0.805	0.493	0.587	0.366	0.613	0.388
	192	**0.417**	*0.277*	0.448	0.282	**0.417**	**0.276**	0.466	0.296	0.598	0.370	0.530	0.293	0.617	0.336	0.756	0.474	0.604	0.373	0.616	0.382
	336	**0.431**	**0.283**	0.473	0.289	0.433	**0.283**	0.482	0.304	0.605	0.373	0.558	0.305	0.629	0.336	0.762	0.477	0.621	0.383	0.622	0.337
	720	**0.463**	**0.301**	0.516	0.307	0.467	0.302	0.514	0.322	0.645	0.394	0.589	0.238	0.640	0.350	0.719	0.449	0.626	0.382	0.660	0.408
	Avg	**0.426**	**0.282**	0.466	0.287	*0.428*	**0.282**	0.481	*0.304*	0.625	0.383	0.550	0.304	0.620	0.336	0.760	0.473	0.610	0.376	0.628	0.379
PEMS03	12	**0.069**	**0.172**	0.072	0.184	*0.071*	*0.174*	0.099	0.216	0.122	0.243	0.090	0.203	0.085	0.192	0.178	0.305	0.126	0.251	0.272	0.385
	24	**0.087**	**0.195**	*0.088*	0.202	0.093	*0.201*	0.142	0.259	0.201	0.317	0.121	0.240	*0.118*	0.223	0.257	0.371	0.149	0.275	0.334	0.440
	48	**0.119**	**0.230**	0.127	0.242	*0.125*	*0.236*	0.211	0.319	0.333	0.425	0.202	0.317	0.155	*0.260*	0.379	0.463	0.227	0.348	1.032	0.782
	96	**0.151**	**0.264**	0.177	0.284	*0.164*	*0.275*	0.269	0.370	0.457	0.515	0.262	0.367	*0.228*	*0.317*	0.490	0.539	0.348	0.434	1.031	0.796
	Avg	**0.107**	**0.215**	0.116	0.228	*0.113*	*0.221*	0.180	0.291	0.278	0.375	0.169	0.281	0.147	0.248	0.326	0.419	0.213	0.327	0.667	0.601
PEMS04	12	**0.078**	*0.185*	0.082	0.197	**0.078**	**0.183**	0.105	0.224	0.148	0.272	0.098	0.218	0.087	0.195	0.219	0.340	0.138	0.262	0.424	0.491
	24	**0.091**	**0.204**	0.094	0.212	*0.095*	*0.205*	0.153	0.275	0.224	0.340	0.131	0.256	0.103	0.215	0.292	0.398	0.177	0.293	0.459	0.509
	48	**0.114**	**0.230**	0.119	0.237	*0.120*	*0.233*	0.229	0.339	0.355	0.437	0.205	0.326	0.136	0.250	0.409	0.478	0.270	0.368	0.646	0.610
	96	**0.141**	**0.254**	0.162	0.275	*0.150*	*0.262*	0.291	0.389	*0.452*	*0.504*	0.402	0.457	0.190	0.303	0.492	0.532	0.341	0.427	0.912	0.748
	Avg	**0.106**	**0.218**	0.114	0.230	*0.111*	*0.221*	0.195	0.307	0.295	0.388	0.209	0.314	0.129	0.241	0.353	0.437	0.231	0.337	0.610	0.590
PEMS07	12	*0.065*	**0.163**	**0.063**	0.171	0.067	*0.165*	0.095	0.207	0.115	0.242	0.094	0.200	0.082	0.181	0.173	0.304	0.109	0.225	0.199	0.336
	24	*0.084*	*0.188*	**0.079**	**0.187**	0.088	0.190	0.150	0.262	0.210	0.329	0.139	0.247	0.101	0.204	0.271	0.383	0.125	0.244	0.323	0.420
	48	*0.103*	*0.210*	**0.100**	**0.203**	0.110	0.215	0.253	0.340	0.398	0.458	0.311	0.369	0.134	0.238	0.446	0.495	0.165	0.288	0.390	0.470
	96	**0.128**	*0.235*	*0.131*	**0.233**	0.139	0.245	0.346	0.404	0.594	0.553	0.396	0.442	0.181	0.279	0.628	0.577	0.262	0.376	0.554	0.578
	Avg	*0.095*	**0.199**	**0.093**	**0.199**	0.101	0.204	0.211	0.303	0.329	0.395	0.235	0.315	0.193	0.271	0.380	0.440	0.165	0.283	0.367	0.451
PEMS08	12	**0.077**	**0.179**	0.091	0.206	*0.079*	*0.182*	0.168	0.232	0.154	0.276	0.165	0.214	0.112	0.212	0.227	0.343	0.173	0.273	0.436	0.485
	24	**0.109**	**0.212**	0.133	0.253	*0.115*	*0.219*	0.224	0.281	0.248	0.353	0.215	0.260	0.141	0.238	0.318	0.409	0.210	0.301	0.467	0.502
	48	**0.177**	**0.232**	0.209	0.249	*0.186*	*0.235*	0.321	0.354	0.440	0.470	0.315	0.355	0.198	0.283	0.497	0.510	0.320	0.394	0.966	0.733
	96	**0.213**	**0.262**	0.492	0.467	*0.221*	*0.267*	0.408	0.417	0.674	0.565	0.377	0.397	0.320	0.351	0.721	0.592	0.442	0.465	1.385	0.915
	Avg	**0.144**	**0.221**	0.231	0.294	*0.150*	*0.226*	0.280	0.321	0.379	0.416	0.268	0.307	0.193	0.271	0.441	0.464	0.286	0.358	0.814	0.659

Table 5. Multivariate time-series forecasting results of iGRU and the baseline models on Weather, Electricity, Exchange, Solar-energy, ETTm1 and ETTm2 datasets. The input length (look-back window) is set to 96 and the prediction length is {96, 192, 336, 720}. The best results are shown in bold, and the second-best results are shown in italics.

Models		**Ours**		**TimeXer**		**iTransformer**		**PatchTST**		**DLinear**		**Crossformer**		**TimesNet**		**TiDE**		**FEDformer**		**Autoformer**	
	Metric	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
Weather	96	*0.167*	0.210	**0.157**	**0.205**	0.174	*0.214*	0.177	0.218	0.196	0.255	*0.158*	0.230	0.172	0.220	0.202	0.261	0.217	0.296	0.266	0.336
	192	0.215	*0.254*	**0.204**	**0.247**	0.221	*0.254*	0.225	0.259	0.237	0.296	*0.206*	0.277	0.219	0.261	0.242	0.298	0.276	0.336	0.307	0.367
	336	0.273	*0.297*	**0.261**	**0.290**	0.278	*0.296*	0.278	0.297	0.283	0.335	*0.272*	0.335	0.280	0.306	0.287	0.335	0.339	0.380	0.359	0.395
	720	0.354	0.349	**0.340**	**0.341**	0.358	*0.347*	0.354	0.348	*0.345*	0.381	0.398	0.418	0.365	0.359	0.351	0.386	0.403	0.428	0.419	0.428
	Avg	*0.253*	*0.278*	**0.241**	**0.271**	0.258	*0.278*	0.259	0.281	0.265	0.317	0.259	0.315	0.259	0.287	0.271	0.320	0.309	0.360	0.338	0.382
Electricity	96	*0.142*	**0.238**	**0.140**	0.242	0.148	*0.240*	0.181	0.270	0.197	0.282	0.219	0.314	0.168	0.272	0.237	0.329	0.193	0.308	0.201	0.317
	192	*0.160*	*0.255*	**0.157**	0.256	0.162	**0.253**	0.188	0.274	0.196	0.285	0.231	0.322	*0.184*	0.289	0.236	0.330	0.201	0.315	0.222	0.334
	336	**0.176**	*0.272*	**0.176**	0.275	*0.178*	**0.269**	0.204	0.293	0.209	0.301	0.246	0.337	0.198	*0.300*	0.249	0.344	0.214	0.329	0.231	0.338
	720	**0.210**	**0.301**	*0.211*	*0.306*	0.225	0.317	0.246	0.324	0.245	0.333	0.280	0.363	*0.220*	*0.320*	0.284	0.373	0.246	0.355	0.254	0.361
	Avg	*0.172*	**0.267**	**0.171**	0.270	0.178	*0.270*	0.205	0.290	0.212	0.300	0.244	0.334	0.192	0.295	0.251	0.344	0.214	0.327	0.227	0.338
Solar	96	*0.194*	*0.243*	**0.187**	0.250	0.203	**0.237**	0.234	0.286	0.290	0.378	0.310	0.331	0.250	0.292	0.312	0.399	0.242	0.342	0.884	0.711
	192	**0.208**	**0.255**	*0.202*	0.271	0.233	*0.261*	0.267	0.310	0.320	0.398	0.734	0.725	0.296	0.318	0.339	0.416	0.285	0.380	0.834	0.692
	336	**0.214**	**0.271**	*0.215*	0.284	0.248	*0.273*	0.290	0.315	0.353	0.415	0.750	0.735	0.319	0.330	0.368	0.430	0.282	0.376	0.941	0.723
	720	**0.214**	**0.264**	*0.220*	0.293	0.249	*0.275*	0.289	0.317	*0.356*	*0.413*	0.769	0.765	0.338	0.337	0.370	0.425	0.357	0.427	0.882	0.717
	Avg	**0.208**	**0.258**	*0.229*	0.274	0.233	*0.262*	0.270	0.307	0.330	0.401	0.641	0.639	0.301	0.319	0.347	0.417	0.291	0.381	0.885	0.711
Exchange	96	**0.086**	0.207	**0.086**	*0.206*	**0.086**	*0.206*	*0.088*	**0.205**	*0.088*	0.218	0.256	0.367	0.107	0.234	0.094	0.218	0.148	0.278	0.197	0.323
	192	*0.181*	*0.304*	0.188	0.308	0.177	**0.299**	**0.176**	**0.299**	**0.176**	0.315	0.470	0.509	0.226	0.344	0.184	0.307	0.271	0.315	0.300	0.369
	336	*0.331*	*0.417*	0.342	0.421	0.331	*0.417*	**0.301**	**0.397**	0.313	0.427	1.268	0.883	0.367	0.448	0.349	0.431	0.460	0.427	0.509	0.524
	720	0.857	0.702	0.870	0.702	*0.847*	**0.691**	0.901	0.714	**0.839**	*0.695*	1.767	1.068	0.964	0.746	0.852	0.698	1.195	0.695	1.447	0.941
	Avg	0.364	0.408	0.372	0.409	*0.360*	**0.403**	0.367	*0.404*	**0.354**	0.414	0.940	0.707	0.416	0.443	0.370	0.413	0.519	0.429	0.613	0.539
ETTm1	96	*0.321*	*0.358*	**0.318**	**0.356**	0.334	0.368	0.329	0.367	0.345	0.372	0.404	0.426	0.338	0.375	0.364	0.387	0.379	0.419	0.505	0.475
	192	*0.364*	**0.382**	**0.362**	*0.383*	0.377	0.391	0.367	0.385	0.380	0.389	0.450	0.451	0.374	0.387	0.398	0.404	0.426	0.441	0.553	0.496
	336	*0.399*	**0.406**	**0.395**	*0.407*	0.426	0.420	*0.399*	0.410	0.413	0.413	0.532	0.515	0.410	0.411	0.428	0.425	0.445	0.459	0.621	0.537
	720	0.470	0.445	**0.452**	*0.441*	0.491	0.459	*0.454*	**0.439**	0.474	0.453	0.666	0.589	0.478	0.450	487	0.461	0.543	0.490	0.671	0.561
	Avg	0.389	*0.398*	**0.382**	**0.397**	0.407	0.410	*0.387*	0.400	0.403	0.407	0.513	0.496	0.400	0.406	0.419	0.419	0.448	0.452	0.588	0.517
ETTm2	96	0.177	0.260	**0.171**	**0.256**	0.180	0.264	*0.175*	*0.259*	0.193	0.292	0.287	0.366	0.187	0.267	0.207	0.305	0.203	0.287	0.255	0.339
	192	0.242	0.304	**0.237**	**0.299**	0.250	0.309	*0.241*	*0.302*	0.284	0.362	0.414	0.492	0.249	0.304	0.290	0.364	0.269	0.328	0.281	0.340
	336	0.306	*0.343*	**0.296**	**0.338**	0.311	0.348	*0.305*	*0.343*	0.369	0.427	0.597	0.542	0.321	0.351	0.377	0.422	0.325	0.366	0.339	0.372
	720	0.408	*0.406*	**0.392**	**0.394**	0.412	0.407	*0.402*	0.412	0.554	0.522	1.730	1.042	0.408	0.403	0.558	0.524	0.421	0.415	0.433	0.432
	Avg	0.283	0.328	**0.274**	**0.322**	0.288	0.332	*0.281*	*0.326*	0.350	0.401	0.757	0.610	0.290	0.333	0.358	0.404	0.305	0.349	0.327	0.371

4. Discussion

Our iGRU model outperforms most baseline models, notably iTransformer, by capturing inter-series relations with GRU modules. Despite the relatively low number of variates in the ETT datasets (seven variates), iGRU achieves a better performance compared to iTransformer and PatchTST models. For example, on the ETTm1 dataset, our iGRU model outperforms iTransformer by more than 4% on average in terms of the MSE metric, highlighting its efficiency in capturing relations between variates and temporal time steps. Our model efficiency is also verified by its performance in datasets with numerous periodic variations, including datasets from traffic, electricity, and PEMS. As observed in Table 4, the iGRU model exhibits noticeably higher MSE and MAE values on the Traffic dataset compared to the PEMSn datasets. This difference can be attributed to the inherent complexity and variability of the Traffic dataset, which includes diverse traffic patterns influenced by external factors such as weather, events, or road conditions, making it less predictable than the PEMSn datasets. In addition, the Traffic dataset's hourly sampling frequency records broader trends, amplifying variability and reducing short-term pattern consistency, whereas the PEMSn datasets' 5 min sampling pattern provides finer patterns, facilitating more predictable temporal dependencies. This trend of elevated errors is consistent across other models evaluated on the traffic dataset, suggesting that the dataset's characteristics pose a general challenge. The results associated with the Traffic, PEMS03, PEMS04, PEMS07 and PEMS08 datasets underscore the capability of iGRU in handling inter-series dependencies more efficiently compared to other baseline models. According to Table 4, iGRU reduces the MSE error (averaged over four prediction lengths) by nearly 6% compared to the second-best baseline (iTransformer) in the PEMS07 dataset, which comprises 883 variates and represents a spatiotemporal type of time-series. iGRU also achieves prediction accuracy comparable to TimeXer while consuming less GPU memory and training time, especially when the prediction length is set to 96. In the traffic dataset, iGRU reduces the MSE error by more than 8.5% on average, relative to TimeXer. When averaged across four prediction horizons, the MSE error also improved by more than 21% in the PEMS08 dataset compared to TimeXer. Additionaly, iGRU performs better than TimeXer in predicting solar power along different forecasting lengths, by more than 9% on average MSE. This highlights the capability of the iGRU model to capture complex cross-variate dependencies. To demonstrate the robustness of our iGRU model, we trained it five times with various random seeds and reported the mean and standard deviation of the results in Table 6. The low standard deviations of the MSE and MAE errors in the test sets associated with different datasets confirm the stability and robustness of the iGRU model.

To illustrate the performance of iGRU on different datasets intuitively, we present a visual comparison of its predictions against the ground truth target series. These visualizations provide a clear and interpretable assessment of the model forecasting accuracy. In the provided plots (Figure 3), the orange line indicates the predictions related to a model and the blue line demonstrates the actual selected sequence. Figure 3 indicates that predictions corresponding to the iGRU are well aligned with the ground truth series on different datasets compared to the predictions generated by the iTransformer.

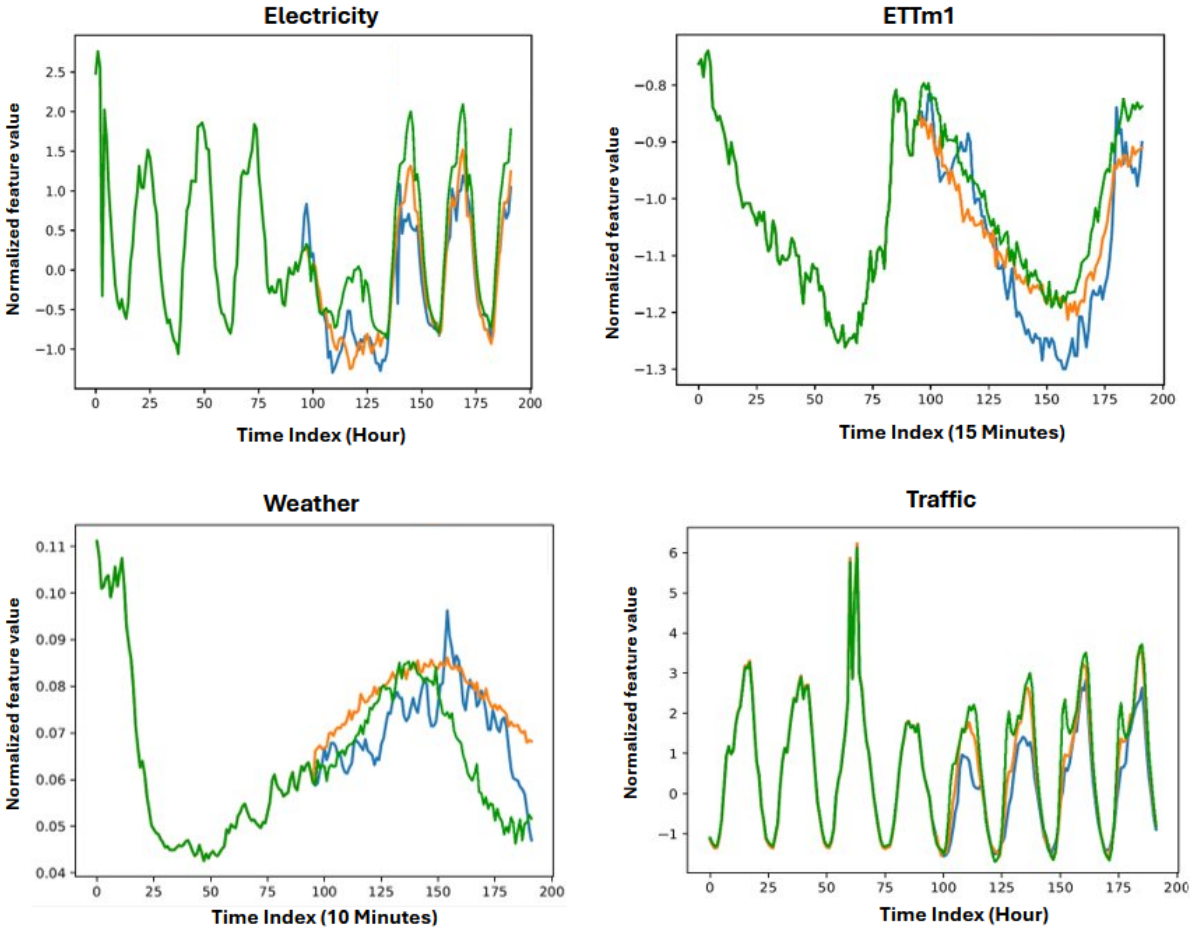


Figure 3. Visual comparison of iGRU and iTransformer on different datasets. The orange line indicates predictions for iGRU and green line corresponds to predictions for iTransformer, with the blue line indicating the ground truth. The look-back window and forecast window lengths are set to 96 for all datasets.

4.1. Model Efficiency

To assess our model's computational efficiency, its memory consumption and training time are compared against those of the other baselines on the Traffic and PEMS07 datasets. Here, by efficiency, we mean training time and GPU memory consumption. Independent runs are conducted using a single A100-40G GPU with the batch size fixed to 16. Our model's efficiency is illustrated in Figure 4, where bubble charts show a visual comparison of efficiency metrics. The vertical axis indicates the prediction MSE, and the horizontal axis depicts the duration of one training iteration (milliseconds/iteration). The bubble size indicates the related memory footprint in Gigabytes (total memory consumption in one epoch). As Figure 4 illustrates, the iGRU model attains the most accurate results, while requiring less or equal training time and memory usage compared to other baselines, except the linear models. DLinear consumes minimal memory and time resources than the other models, while delivering the least accurate forecasts. To further support the claim regarding the computational efficiency of iGRU compared to Transformer-based models, we conducted additional experiments on the Electricity dataset with varying input lengths (96, 336, and 720), while keeping the prediction length fixed at 96. For each configuration, we measured the training time (ms/iteration), peak GPU memory usage, and related model performance (MSE). Figure 5 illustrates how model cost varies across different input lengths. Notably, iGRU consistently maintains a short training time

and low memory footprint, while delivering competitive or better accuracy compared to Transformer-based alternatives. These results confirm that iGRU offers a stable trade-off between efficiency and performance, specifically when the optimal model is sensitive to the length of input window.

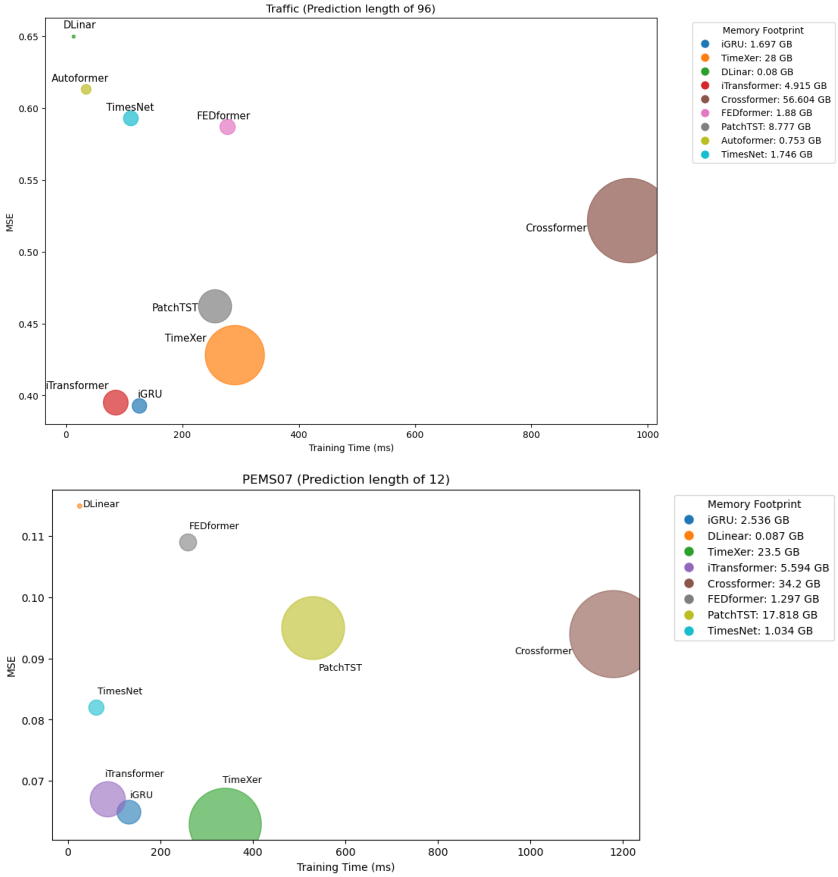


Figure 4. Comparison of iGRU with other baselines in terms of MSE, training time and GPU memory usage on Traffic and PEMS07 datasets. The look-back window is set to 96 and the prediction length is set to 96 and 12 for the Traffic and PEMS07 datasets, respectively.

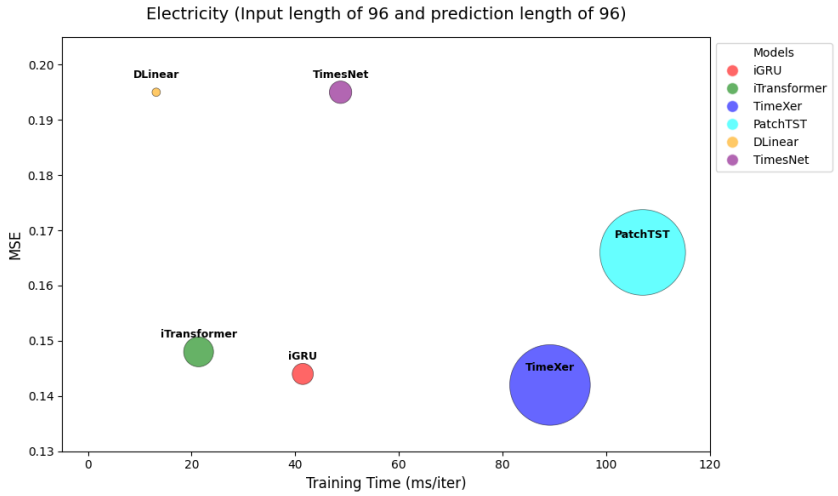


Figure 5. *Cont.*

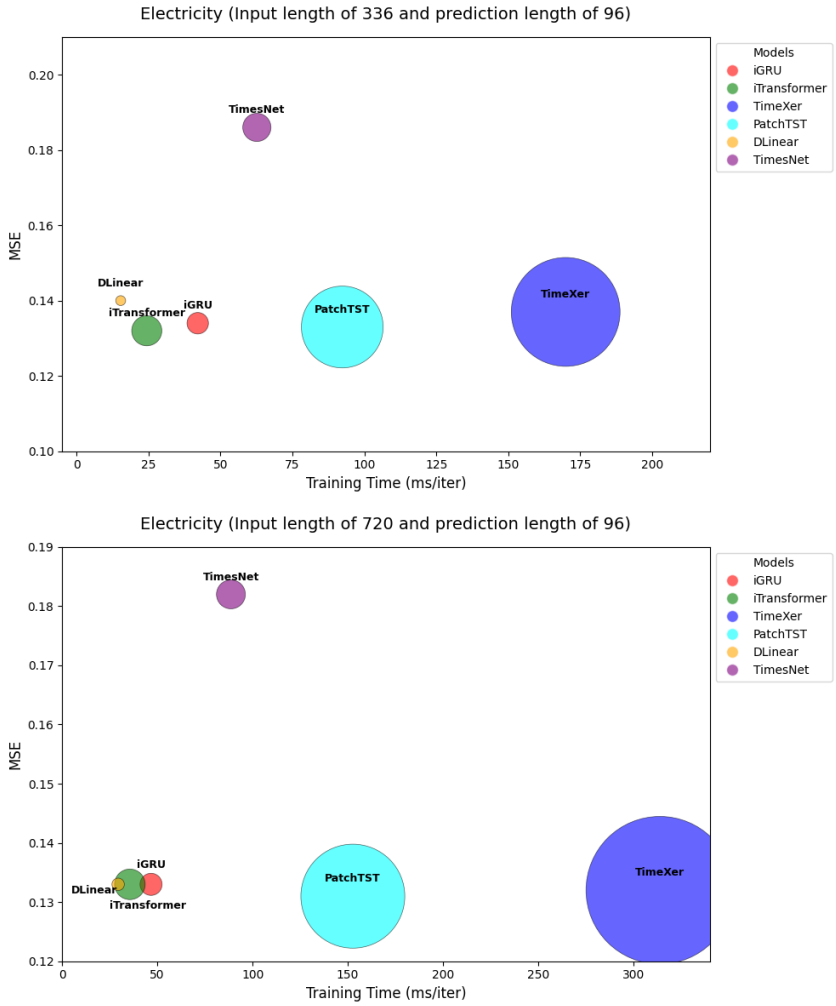


Figure 5. Comparison of training time (ms/iter), model performance (MSE), and memory usage (bubble size) across different input lengths on the Electricity dataset. iGRU consistently demonstrates a low training cost and memory usage while maintaining competitive or better performance compared to Transformer-based models.

4.2. Ablation Study

We demonstrate the importance of the RNN (GRU) and feed-forward layers in our model through conducting experiments with and without the GRU, feed-forward layer and skip connections. The results are reported in Table 7, showcasing the impact of these components on the iGRU performance. The impact of the above components is mostly evident on the Traffic dataset, which is a complex dataset with many time-series variates. The skip connections added to the outputs of the GRU and feed-forward layers contribute to the gradient flow and training efficiency, which helps to attain optimal results. The significant contribution of GRU blocks in capturing cross-variate correlations becomes evident when comparing the results of the model without GRU blocks (W/O GRU) to those of the iGRU model, which incorporates these blocks. This comparison underscores the potential of GRUs to enhance time-series forecasting performance and prompts a raised discussion on whether RNNs, particularly GRUs, should be reconsidered as a powerful tool in time-series analysis, specifically for modeling cross-channel dependencies.

Table 6. Robustness of iGRU model. Five independent runs are conducted with different random seeds.

Dataset	**Electricity**		**Traffic**		**Weather**	
Horizon	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
96	0.142 ± 0.000	0.238 ± 0.000	0.393 ± 0.001	0.267 ± 0.001	0.167 ± 0.002	0.210 ± 0.001
192	0.160 ± 0.000	0.254 ± 0.000	0.416 ± 0.000	0.277 ± 0.001	0.215 ± 0.000	0.254 ± 0.001
336	0.176 ± 0.000	0.272 ± 0.000	0.431 ± 0.000	0.283 ± 0.000	0.273 ± 0.000	0.297 ± 0.000
720	0.210 ± 0.003	0.301 ± 0.003	0.463 ± 0.000	0.301 ± 0.000	0.354 ± 0.001	0.349 ± 0.001
Dataset	**ETTm1**		**ETTm2**		**Exchange**	
Horizon	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
96	0.320 ± 0.001	0.358 ± 0.001	0.178 ± 0.000	0.260 ± 0.000	0.086 ± 0.000	0.207 ± 0.000
192	0.364 ± 0.000	0.382 ± 0.000	0.244 ± 0.001	0.304 ± 0.001	0.181 ± 0.000	0.304 ± 0.000
336	0.398 ± 0.000	0.405 ± 0.000	0.304 ± 0.001	0.343 ± 0.000	0.331 ± 0.000	0.417 ± 0.000
720	0.469 ± 0.001	0.445 ± 0.000	0.408 ± 0.001	0.403 ± 0.001	0.857 ± 0.000	0.702 ± 0.000

Table 7. Ablation of iGRU model without GRU, skip or residual connections and feed-forward layer. The best results are shown in bold.

Design		**W/O F.F**		**W/O F.F + W/O Skip Connection**		**W/O Skip Connection**		**W/O GRU**		**iGRU**	
Metric		**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
ETTm1	96	0.324	0.361	0.325	0.364	0.327	0.364	0.324	0.362	**0.321**	**0.358**
	192	0.366	0.383	0.368	0.387	0.373	0.390	0.367	0.384	**0.364**	**0.382**
	336	**0.399**	**0.405**	0.403	0.410	0.415	0.418	0.400	0.406	**0.399**	0.406
	720	**0.467**	**0.443**	0.475	0.450	0.482	0.455	**0.467**	0.444	0.470	0.445
Traffic	96	0.407	0.281	0.414	0.287	0.502	0.366	0.437	0.282	**0.393**	**0.268**
	192	0.428	0.289	0.437	0.296	0.535	0.372	0.450	0.287	**0.417**	**0.277**
	336	0.445	0.296	0.452	0.303	0.537	0.371	0.464	0.294	**0.431**	**0.283**
	720	0.476	0.313	0.487	0.323	0.572	0.389	0.495	0.312	**0.463**	**0.301**
Weather	96	0.170	0.214	**0.166**	**0.211**	0.169	0.213	0.194	0.232	0.168	**0.211**
	192	0.217	0.256	**0.214**	0.255	0.217	0.257	0.239	0.269	0.215	**0.254**
	336	0.274	0.297	**0.271**	**0.296**	0.277	0.300	0.291	0.307	0.274	0.297
	720	**0.353**	0.349	**0.353**	0.349	0.357	0.352	0.364	0.354	0.354	**0.348**

4.3. Increasing Look-Back Length

Some of the previous Transformer-based works [7,19] have shown that increasing the length of look-back window does not necessarily improve the forecasting results, which can be caused by distracted attention on the increasing (prolonged) input. As the structure of our iGRU model is different from the Transformer-based models, we evaluate the performance of iGRU and its main competitor models, e.g., TimeXer, iTransformer, DLinear and PatchTST in Figure 6 using various input lengths. The results pinpoint the performance promotion of iGRU for longer input windows and its capability to leverage the information over the extended temporal context.

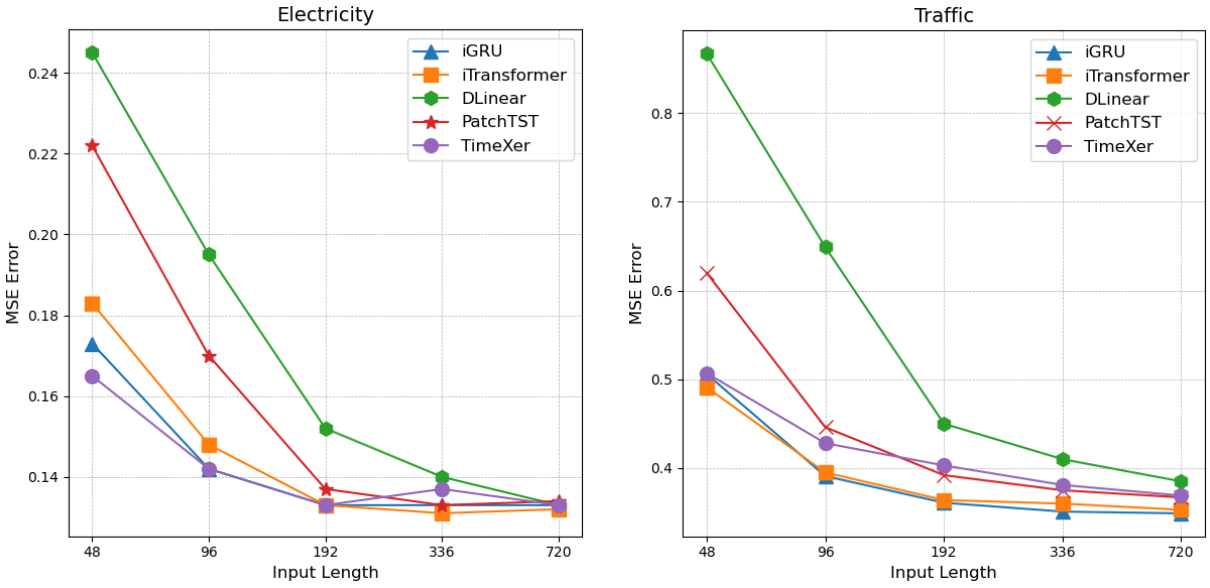


Figure 6. Forecasting with look-back length in {48, 96, 192, 336, 720} and prediction length of 96 on the Electricity and Traffic datasets. The proposed iGRU model exploits enlarged and shortened input lengths and delivers accurate results.

5. Conclusions

In this work, recurrent neural network architecture has been brought back to the time-series forecasting field by using it in a different manner and integrating it with a feed-forward layer. Experimentally, the proposed iGRU achieves results that can compete with those of the other state-of-the-art models, while consuming less training time and memory. In future work, we will investigate large-scale foundation models using the iGRU and perform a more in-depth exploration of time-series analysis.

Author Contributions: Conceptualization, V.N. and A.B.D.; methodology, V.N.; software, V.N.; validation, V.N., A.B.D. and M.B.; formal analysis, A.B.D. and V.N.; investigation, V.N., A.B.D. and M.B.; resources, A.B.D. and V.N.; writing—original draft preparation, V.N.; writing—review and editing, V.N., A.B.D. and M.B.; visualization, V.N.; supervision, A.B.D. and M.B.; project administration, A.B.D.; funding acquisition, A.B.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original data presented in the study are openly available at URL: https://drive.google.com/drive/folders/1ZOYpTUa82_jCcxIdTmyr0LXQfvaM9vIy and https://drive.google.com/drive/folders/1Gv1MXjLo5bLGep4bsqDyaNMI2oQC9GH2 (accessed on 22 April 2025).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chen, Z.; Ma, M.; Li, T.; Wang, H.; Li, C. Long sequence time-series forecasting with deep learning: A survey. *Inf. Fusion* **2023**, *97*, 101819. [CrossRef]
2. Challu, C.; Olivares, K.G.; Oreshkin, B.N.; Ramirez, F.G.; Canseco, M.M.; Dubrawski, A. Nhits: Neural hierarchical interpolation for time-series forecasting. In Proceedings of the 37th AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 6989–6997.

3. Chen, S.-A.; Li, C.-L.; Yoder, N.; Arik, S.O.; Pfister, T. Tsmixer: An all-mlp architecture for time-series forecasting. *arXiv* **2023**, arXiv:2303.06053.
4. Zhou, T.; Niu, P.; Sun, L.; Jin, R. One fits all: Power general time-series analysis by pretrained LM. In Proceedings of the 37th International Conference on Neural Information Processing Systems, New Orleans, LA, USA, 10–16 December 2023; Advances in Neural Information Processing Systems 36 (NeurIPS 2023); Neural Information Processing Systems Foundation, Inc. (NeurIPS): San Diego, CA, USA, 2023; pp. 43322–43355.
5. Luo, D.; Wang, X. Moderntcn: A modern pure convolution structure for general time-series analysis. In Proceedings of the Twelfth International Conference on Learning Representations, Vienna, Austria, 7–11 May 2024; pp. 1–43.
6. Liu, M.; Zeng, A.; Xu, Q.; Zhang, L.; Chen, M.; Xu, Q. Scinet: Time-series modeling and forecasting with sample convolution and interaction. In Proceedings of the 36th International Conference on Neural Information Processing Systems, New Orleans, LA, USA, 28 November–9 December 2022; Advances in Neural Information Processing Systems 35 (NeurIPS 2022); Neural Information Processing Systems Foundation, Inc. (NeurIPS): San Diego, CA, USA, 2022; pp. 5816–5828.
7. Zeng, A.; Chen, M.; Zhang, L.; Xu, Q. Are transformers effective for time-series forecasting? In Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI-23), Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 11121–11128.
8. Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; Long, M. Itransformer: Inverted transformers are effective for time-series forecasting. *arXiv* **2023**, arXiv:2310.06625.
9. Kitaev, N.; Kaiser, L.; Levskaya, A. Reformer: The efficient transformer. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
10. Li, Z.; Qi, S.; Li, Y.; Xu, Z. Revisiting long-term time-series forecasting: An investigation on linear mapping. *arXiv* **2023**, arXiv:2305.10721.
11. Lin, S.; Lin, W.; Wu, W.; Zhao, F.; Mo, R.; Zhang, H. Segrnn: Segment recurrent neural network for long-term time-series forecasting. *arXiv* **2023**, arXiv:2308.11200.
12. Hou, B.-J.; Zhou, Z.-H. Learning with interpretable structure from gated RNN. *IEEE Trans. Neural Networks Learn. Syst.* **2020**, *31*, 2267–2279. [CrossRef] [PubMed]
13. Elsaraiti, M.; Ali, G.; Musbah, H.; Merabet, A.; Little, T. time-series analysis of electricity consumption forecasting using ARIMA model. In Proceedings of the 2021 IEEE Green Technologies Conference (GreenTech), Denver, CO, USA, 7–9 April 2021; pp. 259–262.
14. Zhou, X.; Chen, W.; Wang, Y.X.; Yan, X. Enhancing the locality and breaking the memory bottleneck of transformer on time-series forecasting. In Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019; Advances in Neural Information Processing Systems 32 (NeurIPS 2019); Neural Information Processing Systems Foundation, Inc. (NeurIPS): San Diego, CA, USA, 2019.
15. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21), Virtually, 2–9 February 2021; Volume 35, pp. 11106–11115.
16. Wu, H.; Xu, J.; Wang, J.; Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Online, 6–14 December 2021; Advances in Neural Information Processing Systems 34 (NeurIPS 2021); Neural Information Processing Systems Foundation, Inc. (NeurIPS): San Diego, CA, USA, 2021; pp. 22419–22430.
17. Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; Jin, R. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In Proceedings of the 39th International Conference on Machine Learning PMLR 2022, Baltimore, MD, USA, 17–23 July 2022; pp. 27268–27286.
18. Liu, S.; Yu, H.; Liao, C.; Li, J.; Lin, W.; Liu, A.X.; Dustdar, S. Pyraformer: Low-complexity pyramidal attention for long-range time-series modeling and forecasting. In Proceedings of the Tenth International Conference on Learning Representations (ICLR 2022), Virtual, 25–29 April 2022.
19. Nie, Y.; Nguyen, N.H.; Sinthong, P.; Kalagnanam, J. A time-series is worth 64 words: Long-term forecasting with transformers. *arXiv* **2022**, arXiv:2211.14730.
20. Zhang, Y.; Yan, J. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time-series forecasting. In Proceedings of the Eleventh International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.
21. Das, A.; Kong, W.; Leach, A.; Mathur, S.; Sen, R.; Yu, R. Long-term forecasting with tide: Time-series dense encoder. *arXiv* **2023**, arXiv:2304.08424.
22. Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; Long, M. Timesnet: Temporal 2d-variation modeling for general time-series analysis. *arXiv* **2022**, arXiv:2210.02186.

23. Wang, Y.; Zhang, L.; Chen, M.; Xu, Q.; Zeng, A. Timexer: Empowering transformers for time-series forecasting with exogenous variables. In Proceedings of the Thirty-Eighth Annual Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 9–15 December 2024; Advances in Neural Information Processing Systems 37 (NeurIPS 2024); Neural Information Processing Systems Foundation, Inc. (NeurIPS): San Diego, CA, USA, 2024.
24. Cho, K.; van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.
25. Wang, C.; Liu, Z.; Wei, H.; Chen, L.; Zhang, H. Hybrid deep learning model for short-term wind speed forecasting based on time-series decomposition and gated recurrent unit. *Complex Syst. Model. Simul.* **2021**, *1*, 308–321. [CrossRef]
26. Lawi, A.; Mesra, H.; Amir, S. Implementation of long short-term memory and gated recurrent units on grouped time-series data to predict stock prices accurately. *J. Big Data* **2022**, *9*, 89. [CrossRef]
27. Kim, T.; Kim, J.; Tae, Y.; Park, C.; Choi, J.-H.; Choo, J. Reversible instance normalization for accurate time-series forecasting against distribution shift. In Proceedings of the International Conference on Learning Representations, Vienna, Austria, 4 May 2021.
28. Chen, C.; Petty, K.; Skabardonis, A.; Varaiya, P.; Jia, Z. Freeway performance measurement system: Mining loop detector data. *Transp. Res. Rec.* **2001**, *1748*, 96–102. [CrossRef]
29. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. Pytorch: An imperative style, high-performance deep learning library. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Advances in Neural Information Processing Systems 32 (NeurIPS 2019); Neural Information Processing Systems Foundation, Inc. (NeurIPS): San Diego, CA, USA, 2019.
30. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.

scientific reports

Check for updates

OPEN

A bi-directional cross-channel RNN model for time-series forecasting of dairy production

Vahid Naghashi[1,2], Mounir Boukadoum[1,2] & Abdoulaye Banire Diallo[1,2]✉

Predicting milk production in dairy cattle is essential for precision livestock management, a goal that can be achieved by analyzing historical cow data, including health status, milk quality, and seasonal effects. This challenge is framed as a multivariate time-series forecasting problem, requiring the effective capture of temporal dependencies and interrelationships among dairy-related variables (features). Existing time-series models often struggle to adequately represent these intricate dynamics. To address this, we propose a recurrent neural network (RNN) architecture that leverages Gated Recurrent Units (GRUs) applied bidirectionally along the channel dimension to model these interactions efficiently. The proposed model is further enhanced with feed-forward layers to implicitly capture temporal dependencies. Our approach delivers superior or competitive performance, particularly in terms of various error metrics, compared to state-of-the-art methods when predicting cumulative milk income across different lactation periods. Additionally, it exhibits relatively lower time complexity than recently proposed Transformer and convolution-based models.

Over the past few decades, global food consumption has increased significantly, creating a demand for farmers to enhance their production and farming tools[1]. Dairy farming, as a key agricultural sector, stands out with its substantial contribution to milk production. Recognizing the vital role of the dairy industry in both human life and agriculture, it is of great importance to equip farmers with the latest technologies to achieve efficient milk production[2]. Considering the great success of artificial intelligence in different fields, leveraging deep learning methods to offer reliable predictions regarding calving, diseases, and productivity enables farmers to make informed decisions about their livestock management and implement strategic measures to optimize their herd conditions[3]. The problem of estimating dairy milk yield can be stated as a multivariate time series forecasting task, based on the multiple and heterogeneous dairy features recorded periodically on test days during the lactation months. Dairy features are categorized into health, management, milk quality, milk quantity, farm environmental, and seasonal factors. In this paper, we propose a model to predict milk income throughout the upcoming lactation, taking into account the information from previous lactation period(s). This study builds upon previous works on dairy profitability prediction[2,4], and aims to extend them in several important ways. Prior models primarily focused on a limited lactation period span, often ignoring the heterogeneous nature of dairy production series where milk yield, animal health, management factors, and seasonal effects interact in complex patterns. Despite their success, recently proposed architectures exhibit conceptual limitations in modeling these intricate cross-channel dynamics. Transformer-based models such as PatchTST[5] capture temporal dependencies within each channel by relying on time series patchifying and attention that may not explicitly model the sequential information across channels. MLP-Mixer models[6], while computationally efficient, decompose and mix features across different scales, an operation that might risk obscuring the ordered, non-commutative nature of biological processes. Despite the great performance delivered by State Space Models like S-Mamba[7] in sequential modeling, the potential of simpler architectures like RNNs, when applied in a different manner, has been largely underexplored. This lack of simple models of efficient sequence processing represents a significant gap in current research. To address it, our approach repurposes a Recurrent Neural Network to model dependencies sequentially along the channel dimension. Instead of processing time steps, our model treats the time series variables themselves as an ordered sequence and uses a Gated Recurrent Unit to capture their relationships. This conceptual change allows our model to directly learn the intricate interplay between various dairy like health, management, and production. Our primary contribution is therefore not the invention of a time series model, but the demonstration that a novel application of a classic architecture can outperform more complex state-of-the-art methods in domains where cross-channel dynamics matter. The

[1]Department of Computer Science, Université du Québec à Montréal, 201 Avenue du Président-Kennedy, Montreal, QC H2X 3Y7, Canada. [2]Innovation Chair in Animal Welfare and Artificial Intelligence (WELL-E), Montreal, Canada. ✉email: diallo.abdoulaye@uqam.ca

forecast generated by our model can offer valuable assistance for dairy farmers. Our designed model is based on the well-known Recurrent Neural Network (RNN), specifically the Gated Recurrent Unit (GRU)[8], with modifications to capture cross-channel dependencies among sequences of dairy variables recorded on specific test days across lactation periods. This work makes the following contributions to the state of the art:

- A bidirectional cross-channel model is proposed to predict dairy production from multivariate time series associated with multiple dairy variables recorded along the lactation periods.
- Our designed model captures both temporal and inter-series (cross-channel) correlations, efficiently.
- The proposed model is used to predict milk production factors for dairy cattle in different lactation periods and the results are compared with the baseline models.

Related work

In recent years, the dairy industry has witnessed a proliferation of deep learning-based approaches. The Long-Short Term Memory (LSTM) neural network has been utilized for prediction of milk profitability[2], which relies on an integrated multiple source test-date information to predict profits in future months. A deep learning framework is proposed in[9] to encode dairy features, predict the latent representation of milk yield sequences, and generate the lactation curve by using Convolutional Neural Networks (CNN), encoder, and decoder modules. The prediction of first test day milk yield is accomplished through the application of classical machine learning models, including random forest, in[10]. The authors in[4] proposed a hybrid model based on LSTM and attention modules to predict milk income along the second lactation of dairy cattle. Models for time series forecasting can be generally categorized into conventional statistical methods and deep learning models. Statistical models, such as ARIMA (Auto-regressive Integrated Moving Average) have shown good performance in various forecasting applications[11]. N-BEATS[12] is a well-known deep learning model that is particularly suited for univariate time series forecasting. It uses backward and forward residual connections along with a deep stack of fully connected layers, making it fast to train while also being interpretable. The Transformer architecture has been successful in processing sequences and predicting time-series values. Over the past few years, several time series forecasting models have been proposed that rely on Transformer building blocks. For example, a deep Transformer model with encoder and decoder was introduced in[13] to predict the prevalence of the ifluenza disease. Informer, a Transformer variant with probSparse self-attention in[14] exploits a specialized attention mechanism and a distillation process to emphasize dominant scores, reducing computational overhead and memory usage. In addition, other Transformer based models have been developed, including Autoformer[15], FEDformer[16], Crossformer[17], PatchTST[5], and iTransformer[18]. The well-known PatchTST model[5] pioneered the use of patch embedding and applied self-attention to time series patches, marking a significant advancement in time series forecasting tasks. On the other hand, MLP models utilize linear layers to forecast future values of time series. DeepAR[19] employs MLP in an auto-regressive fashion to make probabilistic predictions. DLinear[20] decomposes time series into trend and seasonality components, uses MLPs to generate the corresponding future trend and seasonality components and then combines them to deliver the final predictions. In another related work, Rlinear[21] introduces a novel method by applying a linear mapping prior to the linear layers, effectively transforming the input space to improve the model's performance in capturing temporal patterns. Recently, mixer models based on MLP layers achieve state-of-the-art results through explicitly decomposing and mixing trend and seasonality components in a hierarchical manner, considering both bottom-up and top-down mixing across multiple temporal scales[6]. By integrating multi-scale information and capturing intricate temporal dependencies with MLP layers, TimeMixer[6] has shown remarkable success. Recently, state space Models have attracted the attention of researchers due to their sequential modeling power and interpretability. One prominent model leveraging SSM is the Mamba model[22], which employs a hierarchical Bayesian approach to represent intricate temporal correlations and learn dynamic patterns over time[22]. More recently, the Mamba model has been utilized to capture cross-channel dependencies in time series and has achieved promising results[7]. However, the potential of RNN models has been overshadowed in time series forecasting domain. Considering the capability of RNNs in sequence modeling, we are interested in exploring the performance of RNNs, specifically GRUs, in multivariate time series forecasting of dairy cattle. In our model, bidirectional GRU modules are employed to capture cross-channel correlations across time series channels, effectively modeling dependencies along both forward and backward directions. The inverted GRU module is integrated with feed-forward layers to represent temporal dependencies, enhancing the model's ability to process time series data.

Methodology

Problem definition

The problem of milk production forecasting can be defined as follows: given a herd of cows of size N and each cow i represented by a set of records $\{\mathbf{x}_j^{(i)} = (x_{j,1}^{(i)}, \ldots, x_{j,T}^{(i)})\}_{j=1..v} \in \Re^{v \times T}$, where $\mathbf{x}_j$ is a sequence of dairy variables, v representing the number of dairy variables considered and T, the record length (total length of the early lactation periods, fixed for all animals), we aim to predict M steps (months) of the upcoming milk production across the next lactation period: $\hat{r}^{(i)} \varepsilon \Re^M$.

RNN and GRU

Recurrent Neural Networks (RNNs) are a specific type of neural network designed for sequential data processing. Unlike traditional feed-forward neural networks, RNNs incorporate loops that allow them to retain a memory of previous inputs by passing information from one step of the sequence to the next, which makes them particularly well suited for tasks involving time series or other sequential data. However, standard RNNs can suffer from issues like vanishing or exploding gradients, which affect their ability to capture long-term

relationships. To address these limitations, advanced RNNs such as Gated Recurrent Units (GRUs)[8] and Long-Short Term Memory (LSTM)[23] variants were developed, introducing update and reset gates to selectively control information flow. In our model (bi-iGRU), we leverage GRUs to effectively capture cross-channel dependencies across time series data, enabling robust representation of intricate interactions between time series channels. The Gated Recurrent Unit (GRU)[8], introduced in 2014, is a well-known type of RNN with a gating mechanism to input or forget information along a sequence of time steps. It utilizes update and reset units to process sequential data in a hidden space. The processing steps of GRU is represented by the following equations:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{1}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{2}$$

$$\hat{h}_t = \phi(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{3}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \tag{4}$$

Where, $\odot$ represents element-wise multiplication. x_t and h_t denote input and hidden state vectors at time t, respectively. z_t, r_t, $\hat{h}_t$ represent the update, reset, and candidate hidden state at time t, respectively. In addition, W_h, W_z, W_r, U_h, U_z, U_r, b_h, b_z and b_r indicate the associated weights and biases, which are learned in an end-to-end manner. GRU has shown promising sequential modeling in various applications while consuming fewer parameters than LSTM[24,25]. In most applications, including time series forecasting, GRUs and LSTMs are commonly used to capture temporal dependencies[26]. However, the ability of GRUs to effectively capture cross-channel dependencies in time series data has often been neglected.

Bi-directional cross-channel GRU for time series forecasting

Time series forecasting using the bi-directional inverted GRU model is depicted in Fig. 1 (in more detail). Bi-iGRU receives an inverted version of the time series[18]. In other words, the input series is first inverted by exchanging its time and channel dimensions. The inverted series is then projected to the model space using a linear mapping layer. RNN models usually process a sequence in one direction, and sometimes sequence processing in one direction does not provide enough insight into the sequential dependencies. Accordingly, in bi-directional iGRU architecture, the inverted series and its reversed version are passed through the GRU blocks, separately, to obtain the bi-directional relations between the time series channels (variates) effectively. We utilize separate GRU blocks in each layer in order to process both the input series and its reversed form. After processing by the corresponding GRU unit, the reversed series is restored to its original order and combined with the processed series through summation. The result is added with the input series using a skip connection and then layer-normalized to stabilize training. We employ a feed-forward layer, which implicitly captures temporal correlations along the model dimension. The feed-forward layer is composed of two linear layers: the

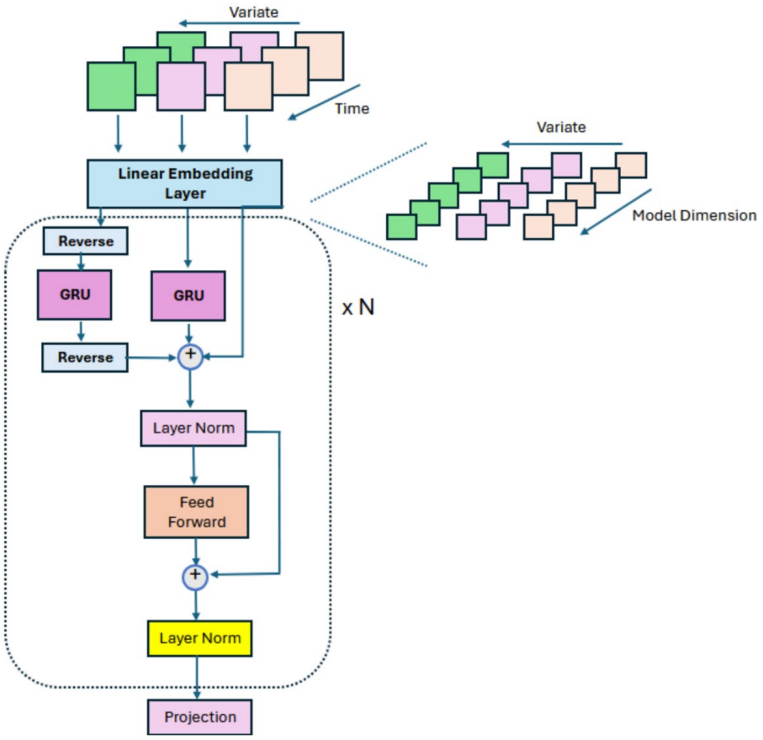


Fig. 1. An illustration of proposed bidirectional cross-channel GRU (Bi-iGRU) model.

first projects the time series from the model dimension to a higher-dimensional feed-forward space (2 times model dimension), followed by a GELU activation function to introduce non-linearity, and the second maps it back to the original model dimension. This enables efficient processing of temporal features while maintaining the representational capacity of the model. The output of the feed-forward layer is combined with its input via a skip connection and then passed to a normalization layer. The final prediction is achieved by feeding the result of the latest step to a projection layer, which is a fully-connected layer that maps the model dimension to the forecasting horizon.

Dataset description

Lactanet, a leading Canadian dairy herd management organization, collects milk production data through routine milk recording, bulk tank sampling, and specialized analyses, adhering to International Committee for Animal Recording (ICAR) guidelines. Our dairy dataset is curated and recorded by the Lactanet organization from a large collection of milk production tests, which were conducted once per month, yielding approximately 2154797 records over six lactation periods of dairy cows, after pre-processing steps. The dataset comprises milk production test records from N=29136 dairy cows across 5019 herds in Quebec, Canada, collected between the years 2006 and 2017. Each test record consists of various features obtained from milk samples and management-related measurements. We pre-processed the dataset by imputing missing values, replacing outliers with meaningful values, and retaining only the records of cows that survived until the end of the sixth lactation. For identifying outliers, we determined values beyond 2 standard deviations (from the mean) as outliers for each variable and marked those extreme values as missing data, then treated them in the missing data imputation steps, replacing them with herd-wise and season-wise averages. To verify that this approach did not distort the data or remove biologically meaningful information, we examined the empirical distributions of the key heavy-tailed variables, including Somatic Cell Count (SCC), Lactose and Milk Urea Nitrogen (MUN). As shown in Fig. 3, more than 97% of the observations fall within the mean $\pm 2\sigma$ range, confirming that the vast majority of biologically plausible records are retained. The remaining extreme values represent highly improbable or erroneous entries. Robust statistical methods (median absolute deviation) were also tried, but we noticed that the critical SCC values were removed by this method. Therefore, we kept the current approach (mean $\pm 2\sigma$) for outlier treatment.

To ensure consistency and completeness, duplicate records were deleted, and inconsistent records were imputed with zeros to maintain dataset size, as removal would reduce records. For instance, the records with negative milk income or negative cumulative milk income were imputed with a default value of zero. Only 0.001% of the records had negative daily milk income or negative cumulative milk income before the replacement phase. Negative daily milk income values were primarily due to data entry errors, while negative cumulative income reflected rare inconsistencies in long-term records. These records were imputed with zeros to ensure data integrity, as negative values for income are biologically implausible for milking cows. Additionally, we removed records associated with the dry period of animals during their lactation periods. For missing data imputation in the dataset, we employed a three-step approach. First, we grouped animals by herd ID, test year, and season, imputing missing values within each group using the mean of each group to which the animal belongs. Next, for any remaining missing values, we imputed them using the mean of the associated herd, based on herd ID. Finally, we addressed any still-unresolved missing values by imputing them with the group mean defined by the test year. The aforementioned three-step imputation approach leverages the hierarchical structure of the dairy dataset to guarantee accurate and contextually relevant replacements for missing values. By first grouping animals by herd ID, test year, and season, local patterns specific to environmental and management conditions are captured, using group means to impute missing information. The subsequent step, imputing with herd-level means, considers broader herd-specific trends, preserving consistency and similarity across animals within the same herd. Finally, using test year averages addresses the remaining gaps with a reasonable replacement based on annual trends. These steps minimize bias by prioritizing the most specific available data while progressively considering broader replacements. At the end, 19 relevant dairy variables were selected and used to construct a multivariate time series based on their recorded variable sequences over six lactation periods (72 months). The selected dairy variables include, daily milk income, cumulative milk income, milking pattern, milking frequency, ANS-CD (animal status code), lactation number, days in milk, milk produced in 24 h (kg), 24-h fat, 24-h protein, lactose, abnormal status, Somatic cell count (SCC), Milk Urea Nitrogen (MUN), test year, test month, test day, test season, and condition code (a sickness indicator). The description of the selected dairy variables is shown in Table 1. In addition, we converted the categorical variables into one-hot encoded versions in order to leverage their corresponding information, more efficiently. The continuous variables were normalized using mean and standard deviation. The distribution of normalized continuous dairy variables before and after outlier treatment is illustrated in Fig. 2. According to Fig. 2 (top), most variables exhibit outlier values in both upper and lower tails, with the SCC factor showing the most extreme values, reaching up to 45. As shown in Fig. 2 (bottom), most variables follow a more symmetric distribution after outlier replacement phase, though lactose retains some extreme values around -20. We further validate our outlier treatment approach through correlation calculation between the dairy variables before and after outlier replacement step. According to Fig. 4, the correlation patterns and similarities remained almost the same, with some correlation scores improved after this step. Post-imputation correlations (Fig. 4 left) show cumulative milk income strongly correlated with fat yield (HR 24 FT) (r = 0.78, vs. 0.71 pre-imputation), consistent with Lactanet's pricing models, and SCC inversely correlated with both daily and cumulative milk income (r = -0.17, -0.10), reflecting udder health impacts. The comparison between correlation scores with and without outliers, highlights that the outlier treatment retains the essential correlations, while removing extreme and unrelated feature values. Finally, we obtained 97 features associated with the time series channels. In our initial assessment of the dairy dataset, we examined the correlations between variables such as milk production, health indicators, milk quality, and seasonality factors. Figure 4 supports our

Variable	Description	Min	Max
Daily Milk Value	Income from milk produced in the test day	0	74.59
Cumulative Milk Value	Cumulative milk income along the lactation months	0	49477
Milking Pattern	Milking pattern of an animal	1	4
Milking Frequency	The milking frequency of the animal for the test (1 time per day, 2 times per day)	0	3
Animal Status Code	Animal status indicator (1 indicates dryness, 2 indicates milking)	1	4
Lactation Number	Lactation number in {1, 2, 3, 4, 5, 6}	1	6
Days in Milk	Number of days that a cow has been milking along the lactation months	0	1575
24-hour-Milk	Milk produced (Kg) in 24 h	0	90.9
24-hour-Fat	Fat in milk produced in 24 h	0	5.802
24-hour-Protein	Protein in milk produced in 24 h	0	3.342
Lactose	Lactose percentage in milk sample (24 h)	0	7.25
Abnormal Status	Quality of the sampled data on the test day	0	8
Somatic Cell Count (SCC)	Somatic cell count in the milk sample	1	25339
Milk Urea Nitrogen (MUN)	Milk Urea Nitrogen (mg/dL)	0	39.9
Test Year	Year indicator of the test date	6	17
Test Month	Month of the test date	1	12
Test Day	Day of the test date	1	31
Test Season	Season of the test	1	4
Condition Code	Code indicator of the animal condition (0 healthy, 1 sick)	0	1

Table 1. Description of dairy variables used to forecast milk income.

hypothesis that cross-channel dependencies are critical for effective time series forecasting of dairy income. For example, daily milk value (income) and cumulative milk income are strongly correlated with 24-h milk, fat, and protein production. Therefore, capturing these critical interactions helps achieve accurate results. We illustrated different steps of our proposed pipeline, including data pre-processing, training, and testing steps, in Fig. 5.

The dataset is randomly split into train and test parts with 80:20 ratio. We created six different versions of the dataset, each corresponding to a different lactation period treated as the forecast horizon. In each case, the input consists of records from one or more previous lactation periods, while the output represents the upcoming lactation period. For instance, in the first version, only the first lactation is used as input, and the second lactation serves as the forecast horizon. In the second version, the input includes data from the first and second lactation periods (covering 24 months), while the third lactation (months 25 to 36) is the forecast horizon. The other four versions follow this pattern, progressively incorporating more lactation periods as input and the next lactation as the forecast window (12 months). We train time series forecasting models with those dairy datasets. After training the models, we have a collection of models to forecast the dairy income along various lactation periods, including lactation 2, 3, 4, 5 and 6. Therefore, we can provide suggestions regarding future profitability to dairy farmers in different lactation periods.

Results and discussion

The bidirectional iGRU model is compared with carefully selected state-of-the-art models, including TimeMixer++[27], S-Mamba[7], TimeMixer[6], iTransformer[18], PatchTST[5], DLinear[20], TimesNet[28], MuMu+Attention[4], and SegRNN[26]. The proposed model is implemented in Pytorch[29] and the experiments were conducted using a single A100-40G NVIDIA GPU. We train our model and other baselines for 15 epochs using Mean Square Error (MSE) loss and the Adam optimizer[30]. We performed hyper-parameter tuning by trying a range of different values, and selecting the value which yielded the best result. In our experiments, the batch size was uniformly selected as 32, and the number of encoder blocks and model dimension were set to 3 and 512, respectively. Two common error metrics were selected to evaluate the models. We assess models both in the forecasting of cumulative milk income and the prediction of the whole dairy channels (97 features):

$$\mathrm{MSE}_{\mathrm{milk}} = \frac{1}{n \cdot T} \sum_{i=1}^{n} \sum_{t=1}^{T} (y_{i,t} - \hat{y}_{i,t})^2 \tag{5}$$

$$\mathrm{MAE}_{\mathrm{milk}} = \frac{1}{n \cdot T} \sum_{i=1}^{n} \sum_{t=1}^{T} |y_{i,t} - \hat{y}_{i,t}| \tag{6}$$

$$\mathrm{MSE}_{\mathrm{total}} = \frac{1}{n \cdot F \cdot T} \sum_{i=1}^{n} \sum_{f=1}^{F} \sum_{t=1}^{T} (x_{i,f,t} - \hat{x}_{i,f,t})^2 \tag{7}$$

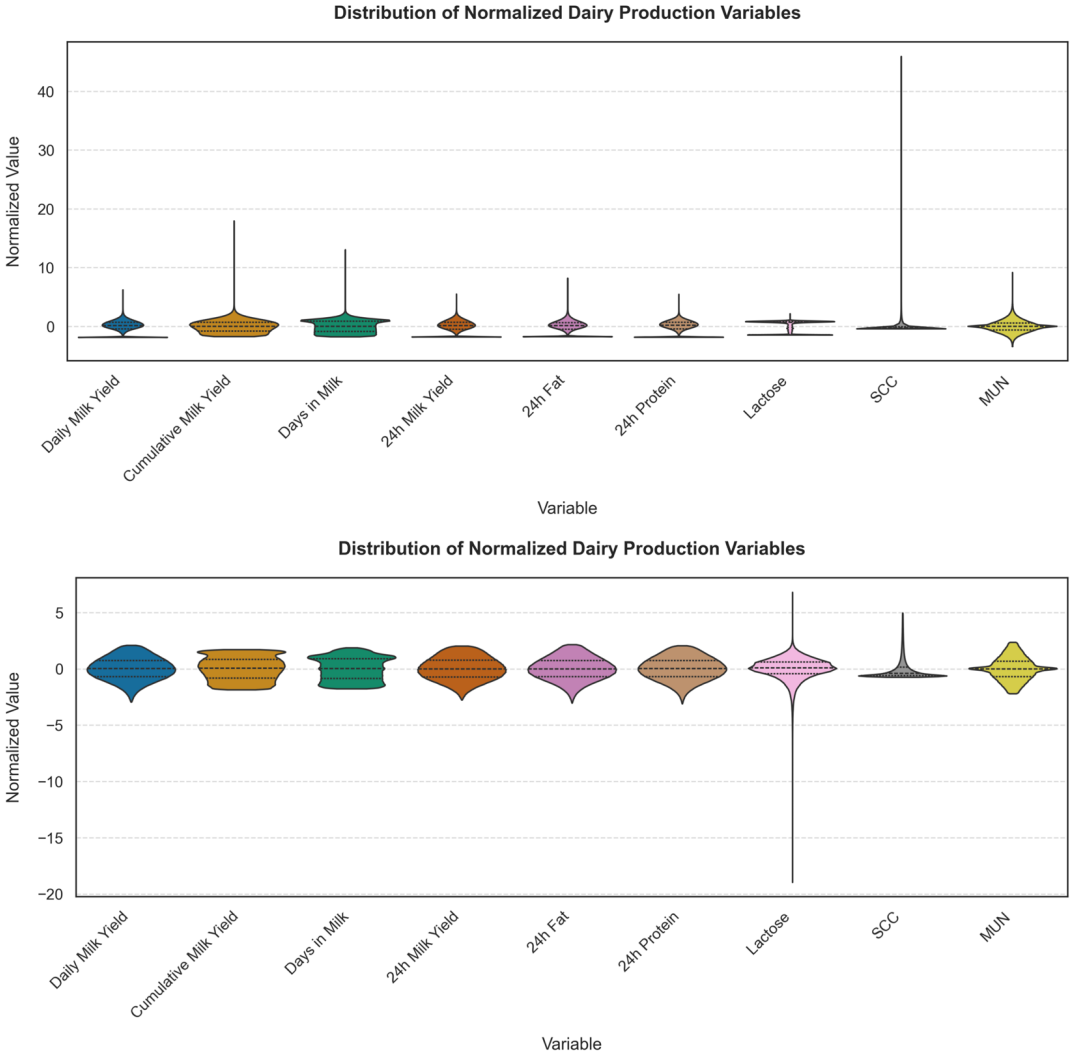
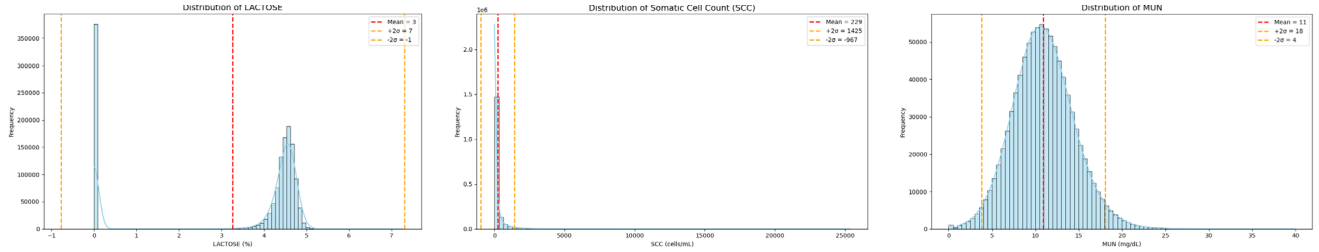
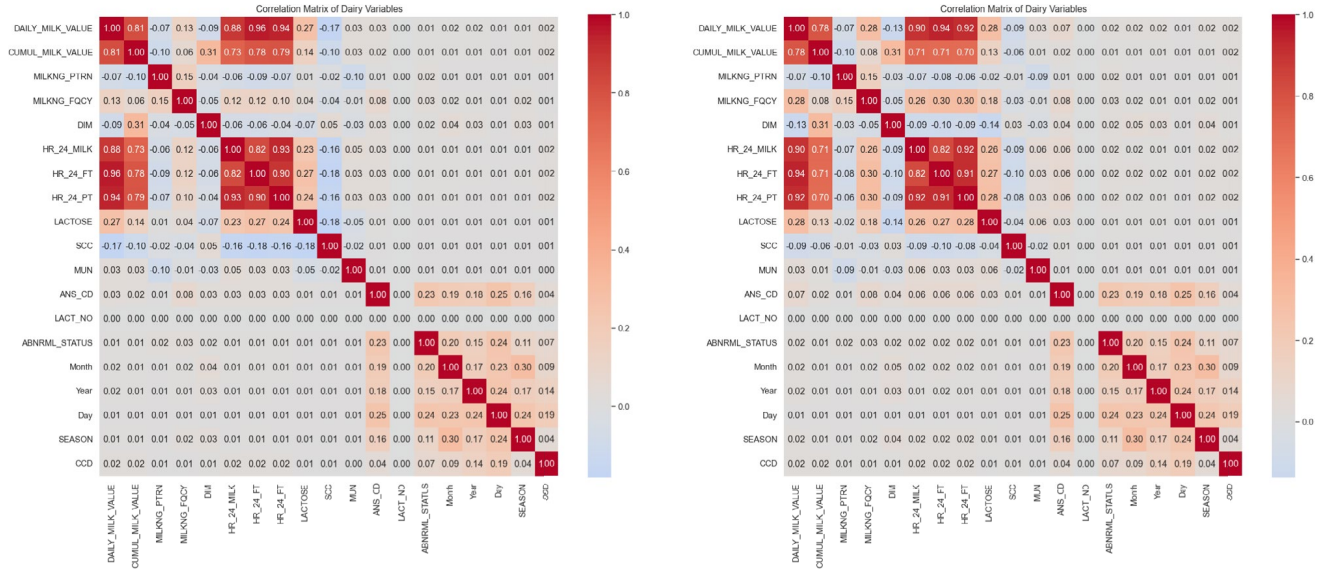


Fig. 2. Violin plots of the selected normalized dairy variables: (top) before and (bottom) after the outlier removal step.



(a) Lactose (%) **(b)** Somatic Cell Count (SCC, cells/mL) **(c)** Milk Urea Nitrogen (MUN, mg/dL)

Fig. 3. Empirical distributions of key milk composition and health indicators. The plots show (**a**) Lactose, (**b**) Somatic Cell Count (SCC), and (**c**) Milk Urea Nitrogen (MUN), along with their mean (red dashed line) and $\pm 2\sigma$ thresholds (orange dashed lines). More than 97% of the data fall within the $\pm 2\sigma$ range, highlighting that biologically plausible records were preserved while extreme outliers were filtered.



(a) Without outliers **(b)** With outliers

Fig. 4. An illustration of the correlations between different dairy variables, with and without outliers.

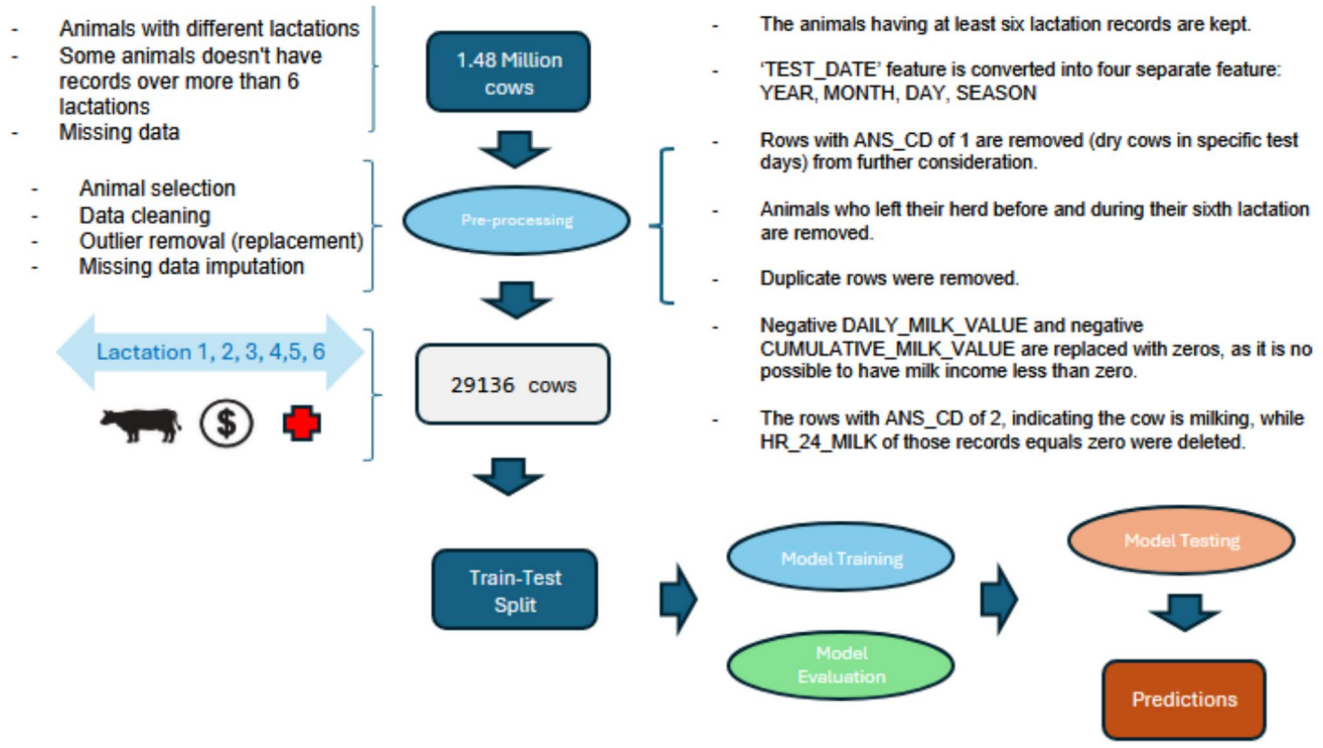


Fig. 5. An illustration of different steps in time series forecasting of dairy production: Pre-processing, model training and testing phases.

$$\mathrm{MAE}_{\mathrm{total}} = \frac{1}{n \cdot F \cdot T} \sum_{i=1}^{n} \sum_{f=1}^{F} \sum_{t=1}^{T} |x_{i,f,t} - \hat{x}_{i,f,t}| \tag{8}$$

Where, n indicates the number of cows, T is the forecast horizon set to 12, F is the number of dairy variables (channels), $y_{i,t}$ represents the actual cumulative milk income for cow i at month t, $\hat{y}_{i,t}$ denotes the predicted cumulative milk income for cow i at month t, $x_{i,f,t}$ is the actual value of feature f for cow i at month t, and $\hat{x}_{i,f,t}$ indicates the predicted value of feature f for cow i at month t.

	1–2		1, 2–3		1, 2, 3–4		1, 2, 3, 4–5		1, 2, 3, 4, 5–6	
Input - Target Lactation	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Bi-iGRU	**0.069**	**0.106**	**0.072**	**0.109**	**0.076**	**0.113**	**0.079**	0.115	**0.086**	0.122
TimeMixer++	0.091	0.112	0.083	0.112	0.085	0.115	0.089	0.119	0.093	0.121
SegRNN	0.088	0.128	0.083	0.124	0.086	0.127	0.089	0.129	0.101	0.136
TimeMixer	0.084	0.136	0.078	0.126	0.080	0.125	0.084	0.127	0.089	0.129
iTransformer	0.125	0.148	0.101	0.132	0.101	0.139	0.105	0.139	0.117	0.150
S-Mamba	0.083	0.124	0.083	0.124	0.089	0.129	0.079	**0.108**	0.087	**0.118**
TimesNet	0.205	0.273	0.114	0.198	0.118	0.209	0.116	0.203	0.115	0.184
DLinear	0.073	0.116	0.075	0.116	0.078	0.117	0.082	0.120	0.087	0.124
PatchTST	0.135	0.162	0.105	0.141	0.103	0.138	0.105	0.138	0.117	0.150

Table 2. Forecasting results of total dairy variates across models and lactation periods. Best results (with the lowest error) are indicated in bold.

	1–2		1, 2–3		1, 2, 3–4		1, 2, 3, 4–5		1, 2, 3, 4, 5–6	
Input - Target Lactation	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Bi-iGRU	**0.134**	**0.260**	**0.134**	**0.258**	**0.144**	**0.264**	0.141	**0.265**	0.227	0.327
TimeMixer++	0.206	0.316	0.150	0.277	0.155	0.277	0.150	0.277	**0.213**	**0.316**
SegRNN	0.196	0.315	0.148	0.277	0.155	0.278	0.149	0.278	0.414	0.442
TimeMixer	0.158	0.291	0.144	0.273	0.150	0.273	0.146	0.275	0.227	0.332
iTransformer	0.242	0.344	0.183	0.306	0.184	0.297	0.195	0.431	0.419	0.431
S-Mamba	0.148	0.277	0.148	0.277	0.149	0.278	**0.140**	0.266	0.229	0.329
TimesNet	0.365	0.422	0.200	0.338	0.209	0.340	0.193	0.328	0.256	0.366
DLinear	0.138	0.270	0.139	0.268	0.144	0.268	0.142	0.272	0.218	0.325
PatchTST	0.325	0.430	0.205	0.343	0.181	0.318	0.180	0.326	0.436	0.493

Table 3. Forecasting results of cumulative milk income across models and lactation periods. Best results (with the lowest error) are indicated in bold.

Model	MSE	MAE	RMSE	R^2
MuMu+Attention	0.0776	0.1190	0.2785	0.5877
Bi-iGRU	**0.0759**	0.1119	**0.2753**	**0.5970**
PatchTST	0.1154	0.1465	0.3394	0.3943
TimesNet	0.1310	0.2086	0.3590	0.2953
TimeMixer++	0.0881	0.1194	0.2966	0.5306
S-Mamba	0.0761	**0.1093**	0.2756	0.5928
SegRNN	0.0890	0.1281	0.2981	0.5271
iTransformer	0.1127	0.1431	0.3323	0.4100
TimeMixer	0.1073	0.1628	0.3275	0.4281
DLinear	0.1176	0.1814	0.3428	0.3733

Table 4. Average model performance metrics for total variates across lactations 2 to 6. Best results are highlighted in bold.

The forecast results for cumulative milk income and total variates are reported in Tables 2 and 3. Tables 4 and 5 show the average of different error metrics across the lactation periods: 2, 3, 4, 5, and 6. According to these tables, Bi-iGRU delivers competitive results in most cases, specifically it achieves better results compared to recently proposed TimeMixer++, TimeMixer, and iTransformer. It also attains competitive or better results relative to S-Mamba and MuMu+Attention models. Bi-iGRU demonstrates a 6% improvement in MSE compared to the TimeMixer++ model when predicting the cumulative production of the fifth lactation period, while requiring less training time. It also outperforms S-Mamba by more than 8% in terms of MSE reduction in the prediction of the sixth lactation period. Bi-iGRU reduces the MSE error by more than 46% compared to the iTransformer model in the prediction of the cumulative income of the sixth lactation. It also demonstrates a 45% reduction in the MSE error compared to the SegRNN model, which is based on an RNN architecture and time series segments. This highlights the superior performance of Bi-iGRU over other RNN-based models. Bi-iGRU excels over TimeMixer++ in forecasting different lactation periods, except the sixth lactation. In particular, it

Model	MSE	MAE	RMSE	R^2
MuMu+Attention	**0.1516**	0.2722	**0.3878**	**0.3821**
Bi-iGRU	0.1518	**0.2712**	0.3880	0.3818
PatchTST	0.2687	0.3867	0.5107	-0.0553
TimesNet	0.2430	0.3571	0.4890	-0.1214
TimeMixer++	0.1889	0.3045	0.4317	0.2073
S-Mamba	0.1550	0.2744	0.3915	0.3712
SegRNN	0.2106	0.3162	0.4481	0.2024
iTransformer	0.2380	0.3306	0.4789	0.0680
Time Mixer	0.2336	0.3442	0.4740	0.0544
DLinear	0.2851	0.4039	0.5267	-0.1475
ARIMA	2.0810	0.9600	1.1720	-12.1898

Table 5. Average model performance metrics for cumulative milk income across lactations 2 to 6. Best results are highlighted in bold.

	Target lact	2	3		4			5				6				
	Input Lact(s)	1	2	1,2	3	2,3	1,2,3	4	3,4	2,3,4	1,2,3,4	5	4,5	3,4,5	2,3,4,5	1,2,3,4,5
MSE	Total	0.070	0.074	0.073	0.078	0.077	0.077	0.082	0.081	0.081	0.082	0.097	0.096	0.093	0.092	0.090
	Cumulative	0.135	0.140	0.135	0.153	0.146	0.145	0.149	0.143	0.142	0.141	0.408	0.400	0.365	0.312	0.258
RMSE	Total	0.264	0.272	0.270	0.279	0.277	0.277	0.286	0.285	0.285	0.286	0.312	0.310	0.305	0.303	0.300
	Cumulative	0.367	0.374	0.367	0.391	0.382	0.381	0.386	0.378	0.377	0.376	0.639	0.633	0.604	0.559	0.508
MAE	Total	0.109	0.112	0.112	0.114	0.114	0.115	0.117	0.117	0.118	0.119	0.129	0.129	0.127	0.127	0.126
	Cumulative	0.261	0.264	0.259	0.273	0.266	0.264	0.274	0.268	0.265	0.265	0.435	0.431	0.415	0.379	0.347

Table 6. Bi-iGRU forecasting results for lactations 2–-6 with varying input lengths. *Total* corresponds to the total dairy forecasting, and *Cumulative* refers to forecasting of the cumulative milk income.

outperforms TimeMixer++ in predicting cumulative income during the second lactation period by more than 34%, and achieves an improvement of 24% in the prediction of total dairy variates, highlighting the efficiency of GRU in capturing cross-channel correlations in forward and backward directions. Bi-iGRU also demonstrates promising performance compared to TimeMixer++ when forecasting the total variates across various lactation periods. Specifically, it achieves improvements of more than 13%, 10.5%, and 11% throughout the third, fourth, and fifth lactation periods, respectively (Table 2). However, Bi-iGRU achieves slightly better results than the MuMu+Attention model with overall MSE improvement of 2% averaged over the lactation periods (Table 4). These results highlight the model's robustness across multiple stages of lactation and its ability to generalize well over extended time horizons (from early lactation to late lactation periods). Consistent gains indicate that Bi-iGRU is an effective model for capturing the underlying temporal and cross-channel dependencies necessary for accurate forecasting in complex real-world dairy datasets.

Dairy forecasting with different input length

Table 6 presents the performance of Bi-iGRU when forecasting milk income across lactation periods 2 to 6, using varying numbers of past lactation periods as context. Here, a clear and prominent trend emerges: The accuracy of the model improves as more historical lactation data is provided. For instance, when predicting the cumulative income during the sixth lactation period, Bi-iGRU achieves an MSE of 0.258 using all five prior lactation records (1–5) as input, compared to an MSE of 0.408, when using only the fifth lactation (a 36.76% improvement). This MSE reduction can be attributed to the bidirectional cross-channel GRU architecture, which effectively leverages the additional context from past lactation periods to capture long-term temporal and cross-channel correlations. For example, variables like cumulative milk income and 24-h fat yield, which are strongly correlated ($r = 0.78$ post-imputation, as shown in Fig. 4), benefit from a longer input horizon that allows the model to better understand their evolving relationship over time. From a practical perspective, this suggests that dairy farmers can achieve more accurate forecasts of future profitability by maintaining comprehensive records over multiple lactation periods. However, the marginal improvement decreases as more lactation records are added (e.g. 17.31% improvement when using 5 prior lactations compared to 4 recent lactation periods), indicating a potential saturation point where additional historical data might yield repetitive temporal patterns.

Interpretability

We analyzed the correlations between predicted cumulative milk income and each feature from the historical time series after training. This post-hoc analysis reveals the importance and impact of each variable in forecasting cumulative milk income. Figure 6 reveals strong positive dependencies with recent production indicators such as cumulative and daily milk value, fat, and protein content, pinpointing that Bi-iGRU effectively prioritizes

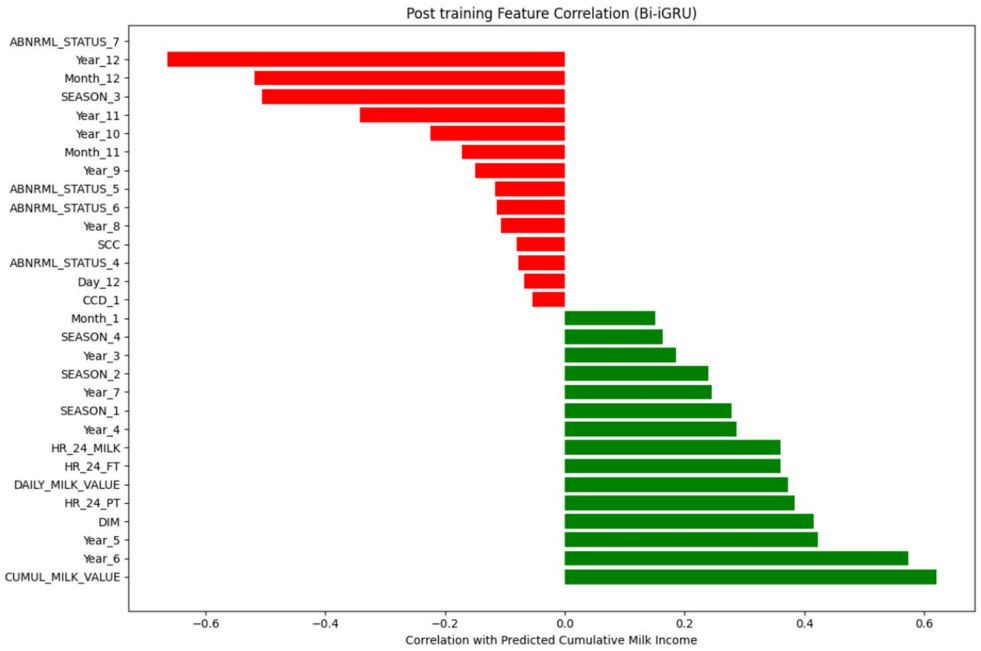
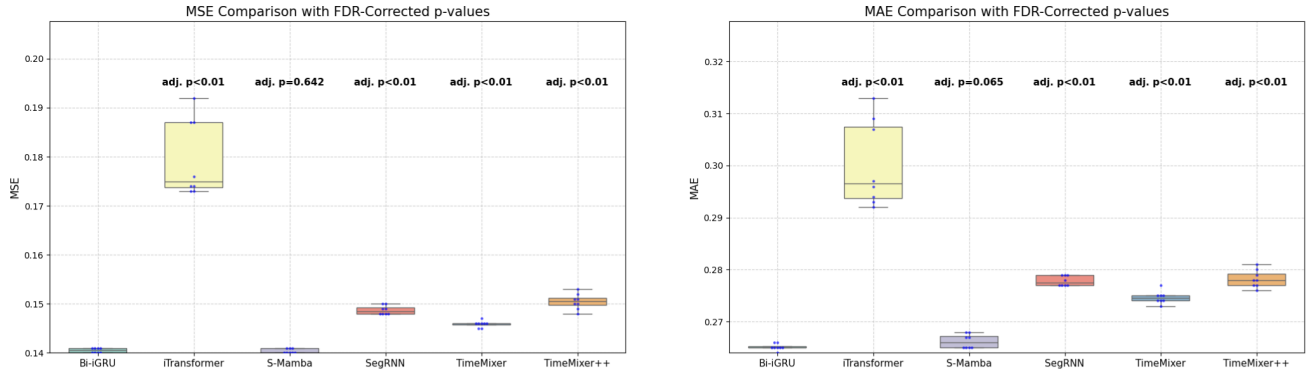


Fig. 6. Correlation between the predicted cumulative milk income with the context dairy variables after model training.



(a) Comparison of model performance in MSE with statistical tests. **(b)** Comparison of model performance in MAE with statistical tests.

Fig. 7. Results of the conducted statistical tests and the relevant box plots (Forecasting of the fifth lactation period).

relevant traits. On the other hand, negative correlations with abnormal status, specific seasonal and year variables suggest that the model captures health and environment-related effects on income. Overall, these patterns align with biological expectations and highlight that forecasts are both data-driven and interpretable.

Statistical significance

To ensure statistical rigor and account for multiple hypothesis testing, pairwise independent t-tests were performed between Bi-iGRU model and other competitive baselines, including iTransformer, S-Mamba, TimeMixer, TimeMixer++, and SegRNN. Figure 7 represents the box plots for MSE and MAE comparisons across these models. To prevent Type I error due to multiple comparisons, the raw p-values were adjusted according to the Benjamini–Hochberg false discovery rate (FDR) correction method. In terms of MSE, Bi-iGRU outperforms most competing models during the fifth lactation period, particularly TimeMixer++ ($t = -16.7332$, adjusted $p < 0.001$), indicating a robust and statistically significant reduction in prediction error. The comparison with S-Mamba, however, shows a nonsignificant difference ($t = 0.4752$, adjusted $p = 0.6420$), indicating comparable performance. For MAE, Bi-iGRU also demonstrates relatively better performance than SegRNN, iTransformer, TimeMixer, and TimeMixer++ (adjusted $p < 0.001$ for all). Although Bi-iGRU slightly outperforms S-Mamba, the difference remains non-significant after correction (adjusted $p = 0.065$). These results confirm that Bi-iGRU achieves statistically significant improvements in both MSE and MAE compared to most baselines, and that

Design		W/O feed-forward		Unidirectional GRU		W/O GRU		Bi-iGRU	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Lactation 6	Total	0.092	0.128	0.095	0.128	0.118	0.168	**0.089**	**0.125**
	Cumulative	0.327	0.393	0.402	0.423	0.430	0.443	**0.242**	**0.338**
Lactation 5	Total	0.082	0.119	0.083	0.119	0.105	0.161	**0.082**	**0.119**
	Cumulative	**0.141**	0.266	0.144	**0.264**	0.173	0.292	0.142	0.265

Table 7. Ablation study of Bi-iGRU model by removing the feed-forward layer, GRU modules and two-directional channel-wise GRU. The results with the lowest errors are shown in bold.

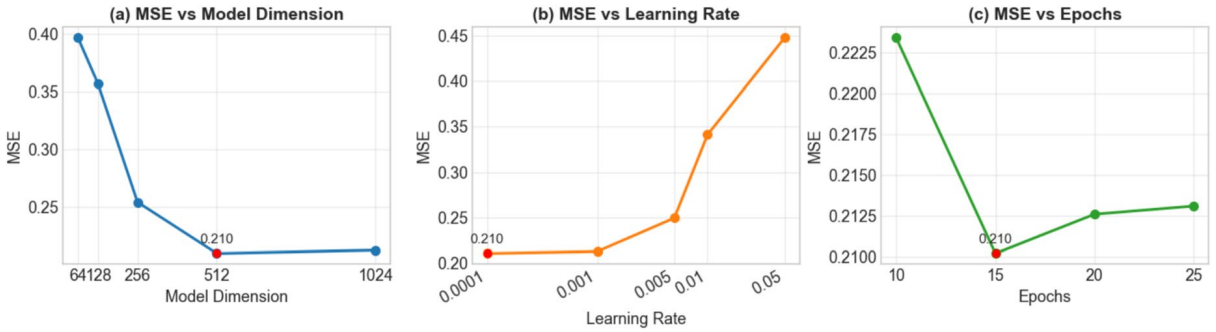


Fig. 8. Hyperparameter sensitivity analysis of the proposed model.

these findings remain robust even after applying multiple-testing correction, underscoring the reliability of the results.

Ablation study

The ablation study in Table 7 evaluates the impact of removing key components of the Bi-iGRU model, namely the feed-forward layer, the bidirectional mechanism and the GRU modules, on the prediction performance for the fifth and sixth lactation periods. When the feed-forward layer is removed (W/O Feed-Forward), the MSE for cumulative milk income in the sixth lactation increases from 0.242 to 0.327 (a 26% increase), highlighting the critical role of the feed-forward layer in capturing temporal dependencies along the model dimension. This layer, composed of two linear transformations with a GELU activation, introduces non-linearity that boosts the model's ability to capture complex temporal patterns, such as the gradual increase in cumulative milk income over lactation months. Similarly, switching to a unidirectional GRU (Unidirectional case) results in an MSE of 0.402 for the sixth lactation, a 40% increase compared to Bi-iGRU, underscoring the importance of processing the time series in both directions to capture cross-channel relations comprehensively. Finally, removing the GRU modules entirely (W/O GRU) leads to the highest cumulative income MSE (0.430), indicating that the GRU's gating mechanism is essential for selectively retaining and forgetting information across time series channels. These experiments validate the synergistic design of Bi-iGRU, where each component contributes to its overall performance. We further studied hyper-parameter sensitivity of our model by conducting experiments using different values of the key hyper-parameters. Figure 8 illustrates the test error under different configurations. The model dimension, learning rate, and number of training epochs were varied and the mean squared error (MSE) was calculated in each setting. The model achieves the best performance at a dimension of 512, learning rate 0.001, and 15 epochs. Bi-iGRU model was trained and tested with different feature subsets, including health, milk quality, and management features on prediction of the cumulative income along the sixth lactation period. We included historical cumulative milk income in all cases. Results (Fig. 9) show that seasonal features alone produce the best performance (MSE=0.2165, MAE=0.3167, R2=0.4456), followed by the milk quality, management and health subset.

Visualization

We intuitively demonstrate the superior performance of Bi-iGRU by visualizing the predicted versus ground truth cumulative milk income and daily milk production during different lactation periods. Figures 10, 11, 12 and 13 provide visual comparisons of Bi-iGRU predictions against baseline models for daily and cumulative milk income in different lactation periods. Figure 10 illustrates the prediction of daily milk income during the second lactation, where Bi-iGRU closely follows the ground truth trend, specially during the initial months, where a sharp decrease is observed before stabilization. In contrast, models like TimeMixer and iTransformer underestimate the target values, failing to adapt to the rapid decline. Figure 11, which shows cumulative milk income for the fourth lactation period, further highlights Bi-iGRU's accuracy, with predictions aligning closely with the ground truth, while baselines like iTransformer exhibit larger deviations, particularly during the last months, possibly due to their reliance on channel-wise attention mechanisms that might overlook fine-grained

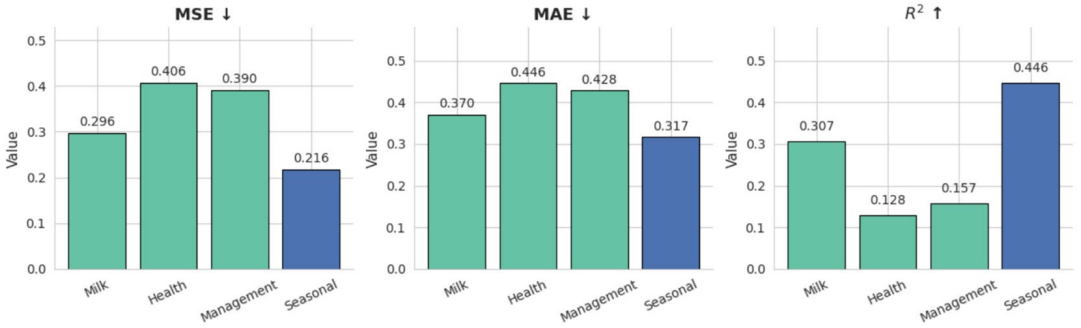


Fig. 9. Bi-iGRU performance using different feature subsets on cumulative milk income prediction during the sixth lactation. Seasonal features yield the best overall results.

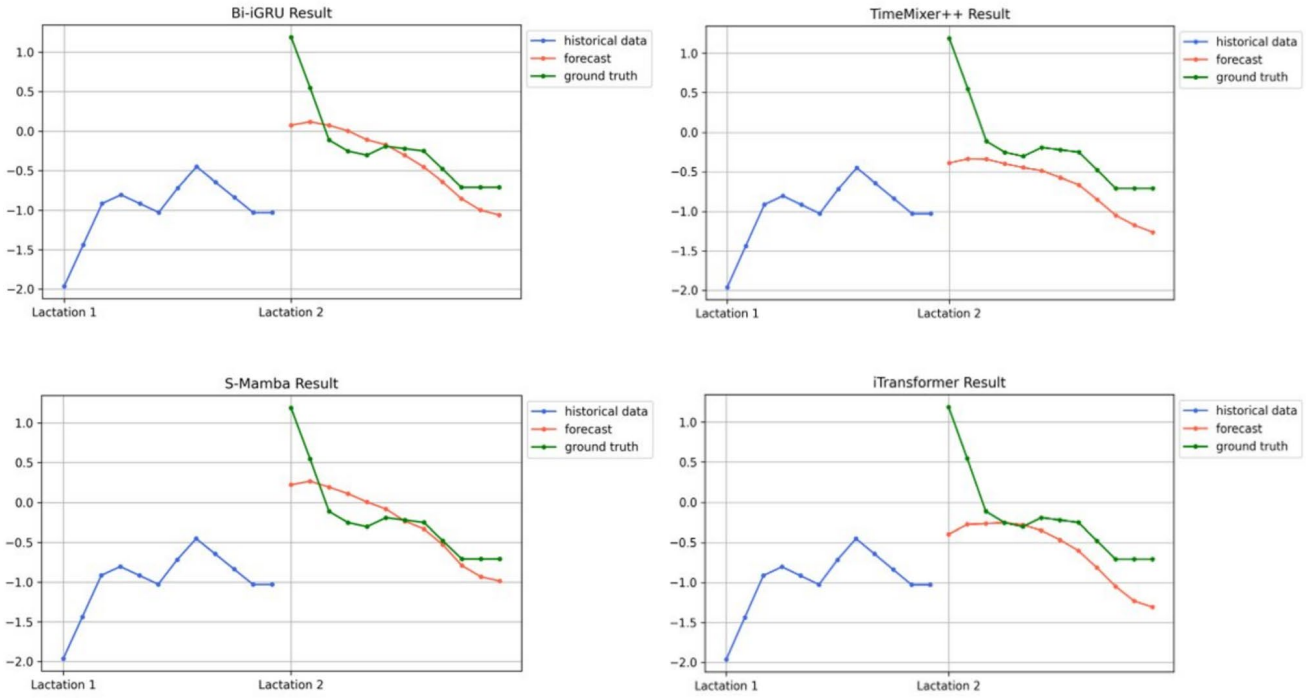


Fig. 10. Visual comparison of prediction results among different models for forecasting daily milk income during the second lactation.

dependencies. Figure 12, focusing on daily milk income during the fourth lactation, reveals that Bi-iGRU and S-Mamba achieve the most accurate predictions, with Bi-iGRU slightly outperforming S-Mamba in capturing the subtle fluctuations in income, such as the variations in the beginning and end of the lactation period. Finally, Fig. 13 demonstrates Bi-iGRU's superior performance in forecasting cumulative milk income during the sixth lactation, particularly in the final months, where its delivered predictions remain closely aligned with the ground truth, unlike other models that diverge significantly. These accurate predictions in later lactation months are critical for dairy farmers, as they facilitate reliable long-term planning, such as estimating total income at the end of a lactation cycle to make informed decisions about herd replacement or investment in farm infrastructure.

Model efficiency

To verify the computational efficiency of our model, we compare the memory consumption and training time with those of the other baselines. Independent runs were performed using a single A100-40G GPU with the batch size fixed at 32. The efficiency of our model is illustrated in Fig. 14, where bubble charts display a visual comparison of efficiency metrics. In this Figure, the vertical axis indicates the MAE of the predicted cumulative milk income, and the horizontal axis shows the duration of one training iteration. The bubble size represents the peak memory footprint in one epoch (in gigabytes). As Fig. 14 illustrates, Bi-iGRU sacrifices a small amount of memory and time to achieve a competitive improvement in accuracy, with the lowest MAE (0.265). On the other hand, DLinear is the fastest model, with the shortest train time (3.05 ms/iter), while achieving relatively low accuracy among the other models. SegRNN, S-Mamba, TimesNet, TimeMixer, and TimeMixer++ show

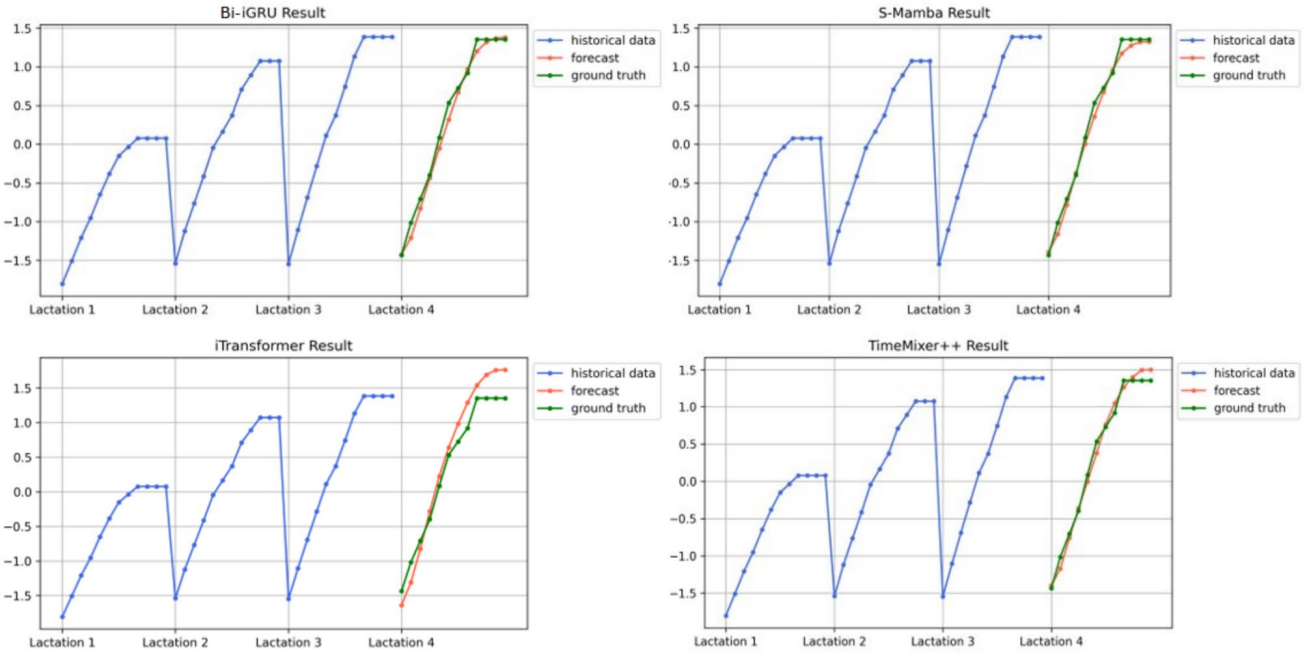


Fig. 11. Visual comparison of prediction results among different models for forecasting cumulative milk income during the 4th lactation.

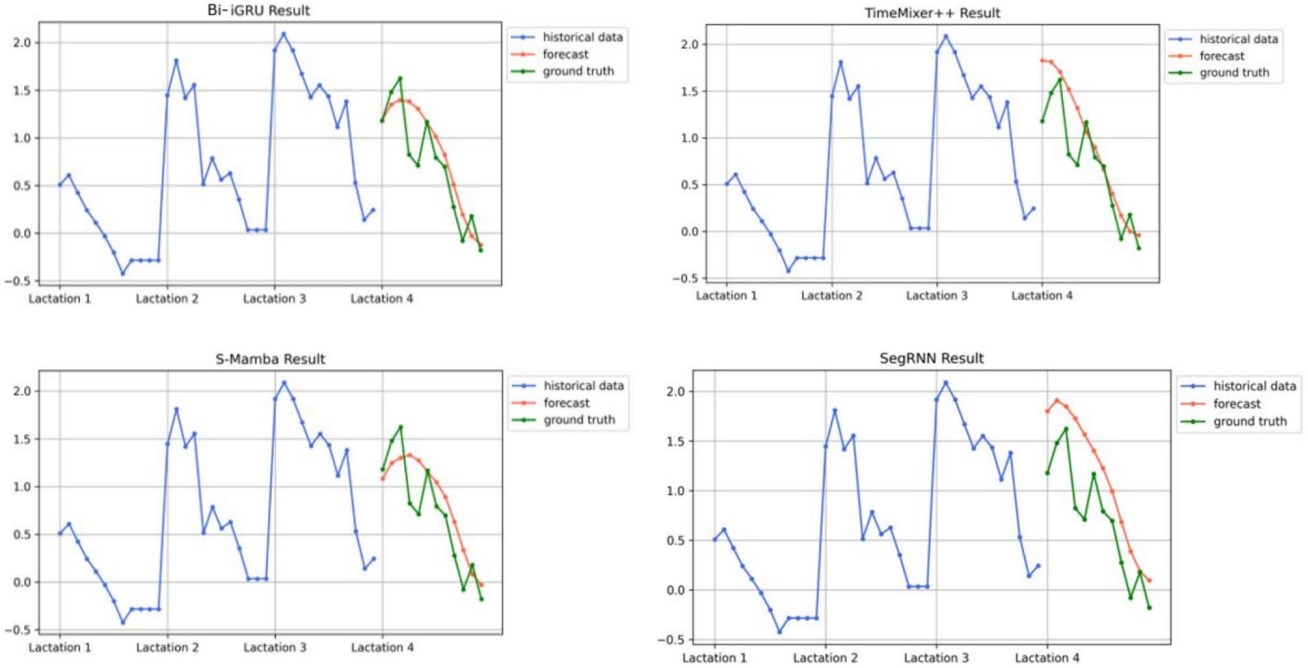


Fig. 12. Visual comparison of prediction results among different models for forecasting daily milk income during the 4th lactation.

higher error, with TimeMixer++ being slower than others while consuming less memory. PatchTST demands high memory and training time compared to the other models. iTransformer requires similar memory and train time as Bi-iGRU, while delivering the worst result (MAE of 0.431). In addition, we report parameter count, FLOPs and training time for each model in Table 8. Although Bi-iGRU has the largest number of parameters, its recurrent architecture allows for parameter reuse across time steps, resulting in lower FLOPs and faster training compared to models with fewer parameters but more computationally intensive operations, such as self-attention or 2-dimensional convolutions. Overall, Bi-iGRU offers an optimal balance of accuracy and efficiency, with S-Mamba achieving the second-best performance.

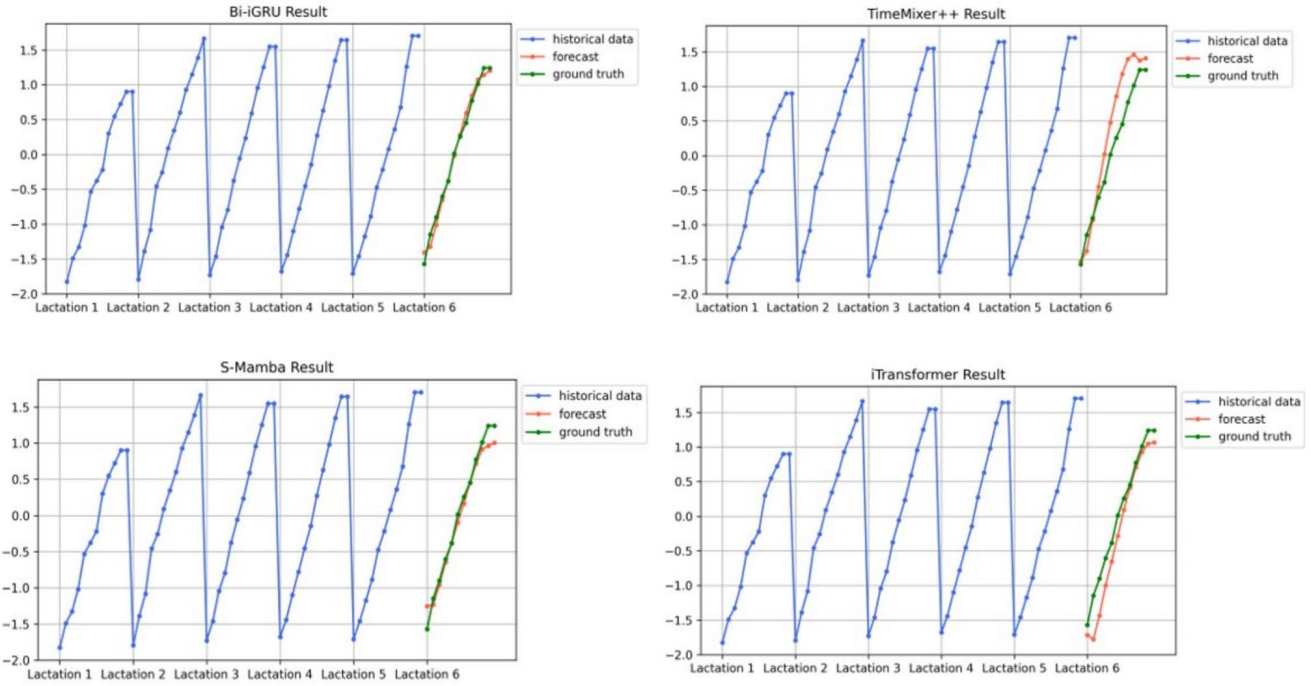


Fig. 13. Visual comparison of prediction results among different models for forecasting cumulative milk income during the 6th lactation.

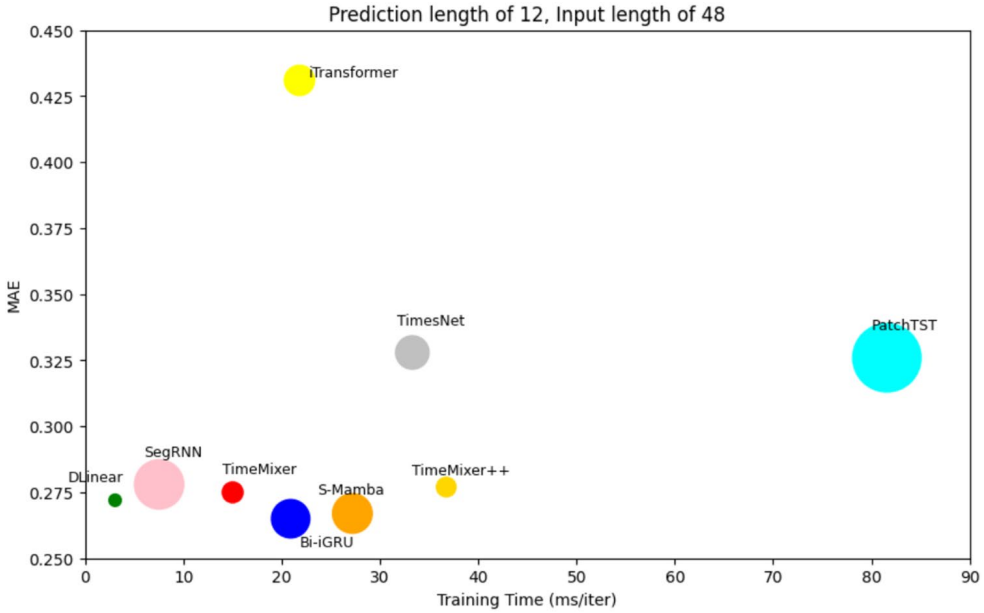


Fig. 14. Comparison of model efficiency in terms of peak GPU memory and training time.

Model stability

Figure 15 illustrates the distribution of MSE and MAE across independent runs for Bi-iGRU and baseline models, forecasting the fifth lactation period. Bi-iGRU's violin plot (highlighted in blue) depicts a narrow distribution with a mean MSE of 0.141 and a low variance, indicating high stability and consistency in its predictions. The internal quartiles reveal that 75% of Bi-iGRU's MSE values fall below 0.145, a threshold that iTransformer (mean MSE: 0.195) and SegRNN (mean MSE: 0.149) fail to achieve, as their distributions are wider and shifted towards higher errors. This stability is likely due to the layer normalization and skip connections in Bi-iGRU's architecture (Section 3.3), which stabilize training. For dairy farmers, this consistency across various runs translates to reliable forecasts that can be trusted for critical decisions, such as adjusting milking frequency or monitoring SCC to prevent mastitis (a health issue reflected by high SCC values). The narrow error distribution

Model	Params	FLOPs	Training time (ms/iter)
Bi-iGRU	11.07 Million	1076.81 MFLOPs	20.92 ms
MuMu+Attention	4.212 Million	174.52 MFLOPs	14.70 ms
S-Mamba	6.86 Million	1350 MFLOPs	27.20 ms
PatchTST	1.62 Million	3740 MFLOPs	81.58 ms
SegRNN	1.2 Million	926 MFLOPs	7.54 ms
TimeMixer	0.07 Million	566.17 MFLOPs	15.02 ms
DLinear	0.0012 Million	0.12 MFLOPs	3.06 ms
iTransformer	4.757 Million	461.18 MFLOPs	21.82 ms
TimeMixer++	1.18 Million	18538.53 MFLOPs	36.75 ms
TimesNet	3.53 Million	633.33 MFLOPs	33.30 ms

Table 8. Comparison of models: parameters, FLOPs, and training time.

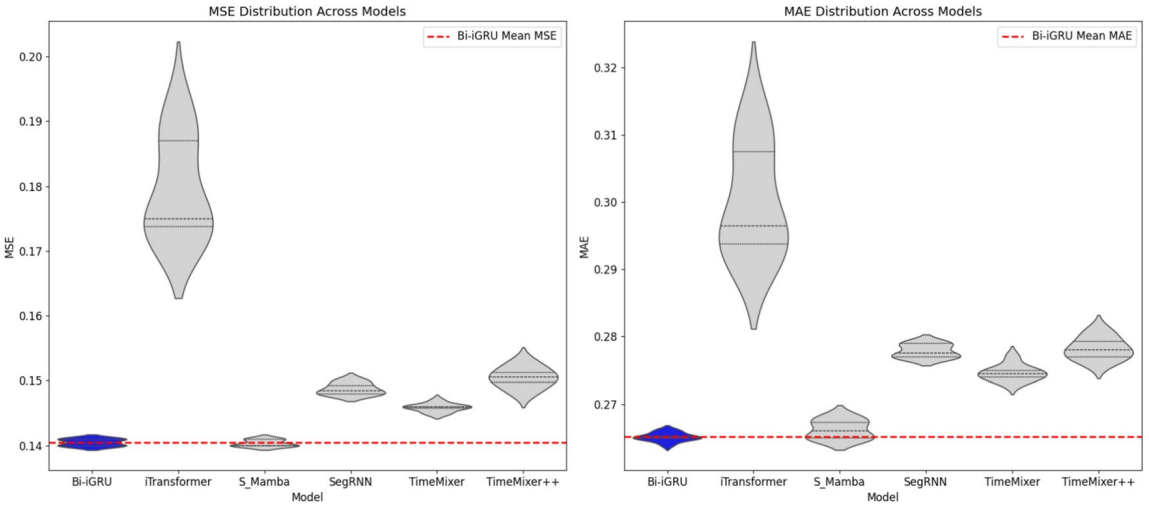


Fig. 15. Violin plots for illustration of robustness of our model (Forecasting of the 5th lactation period.).

also underscores that Bi-iGRU is less sensitive to variations in the dataset, such as those introduced by the outlier treatment phase, making it a robust choice for real-world applications where data quality may vary.

Economic and decision-making interpretability

Bi-iGRU achieved a competitive error (MAE of 0.260), when predicting the cumulative income during the second lactation period. Differences in error metrics can be translated into economic terms using the observed variability of cumulative milk income (2225 CAD per cow per month), which was calculated based on the actual cumulative milk income in the dataset. Therefore, the improvement achieved by our model corresponds to approximately 120–380 CAD per cow per month compared with Transformer-based models such as iTransformer and PatchTST. At the herd level (considering herd of 150 cows), this represents an estimated saving of 0.22-–0.68 million CAD, annually. More accurate forecasts of cumulative milk income and their economic interpretation enable producers to assess expected cash flow ahead of time during the lactation period, which supports proactive adjustments in feeding strategies, calving schedules, and herd management planning. For instance, a predicted decline in income allows managers to reallocate feed resources, implement preventive health interventions, or change milking frequency to stabilize production and profitability. The associated economic improvements are reported in Tables 9 and 10.

Bi-iGRU demonstrated competitive performance in forecasting sixth lactation milk income. Relative to SegRNN, iTransformer, and PatchTST, the observed MAE reduction (0.12–0.18) corresponds to 260–396 CAD per cow per month, representing an estimated saving of 0.46–0.71 million CAD annually for a herd of size 150 animals. These economic gains are derived from more accurate long-horizon forecasts of cumulative milk income, which allow improved resource allocation. Compared to the best competing model (MuMu+Attention), the difference in MAE was economically marginal (2.23 CAD per cow per month, indicating that Bi-iGRU maintains a competitive level of efficiency).

Conclusion

In this study, we developed a time series forecasting model tailored for dairy production and verified its effectiveness through comprehensive experiments. The results pinpoint its capability to efficiently capture and

Model	MAE	ΔMAE (vs Bi-iGRU)	Improvement (CAD/cow/month)	Annual impact (CAD, 150 cows)
Bi-iGRU	0.260	–	–	–
MuMu+Attention	0.257	-0.003	$ -6.68	$ -12,015
TimeMixer++	0.316	$+0.056$	$124.6	$224,280
SegRNN	0.315	$+0.055$	$122.4	$220,320
TimeMixer	0.291	$+0.031$	$69.0	$124,200
iTransformer	0.344	$+0.084$	$187.9	$338,220
S-Mamba	0.260	0.000	$0.0	$0
TimesNet	0.422	$+0.162$	$360.5	$649,000
DLinear	0.270	$+0.010$	$22.3	$40,140
PatchTST	0.430	$+0.170$	$378.3	$681,000

Table 9. Economic interpretation of predictive improvements for Lactation 2. MAE error values are translated into Canadian dollars (CAD) using $\sigma_{\text{income}} = 2225.38$. Annualized herd impact assumes 150 cows and 12 months.

Model	MAE	ΔMAE (vs Bi-iGRU)	Improvement (CAD/cow/month)	Annual Impact (CAD, 150 cows)
Bi-iGRU	0.315	–	–	–
MuMu+Attention	0.314	-0.001	$\$ - 2.23$	$\$ - 4{,}005$
TimeMixer++	0.316	$+0.001$	$2.23	$4,005
SegRNN	0.442	$+0.127$	$282.6	$508,680
TimeMixer	0.332	$+0.017$	$37.8	$68,040
iTransformer	0.431	$+0.116$	$258.2	$465,000
S-Mamba	0.318	$+0.003$	$6.68	$12,024
TimesNet	0.366	$+0.051$	$113.5	$204,300
DLinear	0.325	$+0.010$	$22.3	$40,050
PatchTST	0.493	$+0.178$	$396.1	$712,980

Table 10. Economic interpretation of predictive improvements for Lactation 6. Normalized MAE values are converted to Canadian dollars (CAD) using $\sigma_{\text{income}} = 2225.38$. Annualized herd impact assumes 150 cows and 12 months.

represent both temporal and cross-channel dependencies, achieving linear time complexity with respect to the number of time series channels. Leveraging a bidirectional GRU-based architecture, our model delivers accurate predictions for cumulative milk income, milk yield, and other dairy factors, empowering dairy farmers with valuable insights to optimize decision-making across lactation periods ahead of time. Furthermore, the model's design offers adaptability, accommodating diverse dairy datasets and potentially extending to other agricultural forecasting scenarios. By bridging deep learning with practical agricultural demands, this paper provides a baseline for future enhancements, such as refining predictions with multi-modal data, to further support sustainable and profitable dairy farming practices. Although Bi-iGRU demonstrates promising predictive performance on the Lactanet dataset, its generalizability to other regions, breeds, or management conditions remains to be validated and this represents one of the limitations of the current work. The model is designed to capture general temporal and inter-feature patterns, which suggests transferability to other datasets associated with different regions and even datasets from other domains. Future work will focus on external validation, and domain adaptation to ensure robust performance on more recent dairy datasets and analyzing broader geographic and production contexts.

Data availability

Data will be available upon request. Please contact Abdoulaye Banire Diallo: diallo.abdoulaye@uqam.ca

Received: 17 September 2025; Accepted: 13 November 2025
Published online: 21 November 2025

References

1. Bruinsma, J. *World Agriculture: Towards 2015/2030: An FAO Study* (Routledge, 2017).
2. Frasco, C. G. *et al.* Towards an effective decision-making system based on cow profitability using deep learning. In *Proceedings of the 12th International Conference on Agents and Artificial Intelligence*, vol. 2, 949–958 (2020).

3. Borchers, M. R. et al. Machine-learning-based calving prediction from activity, lying, and ruminating behaviors in dairy cattle. *J. Dairy Sci.* **100**, 5664–5674 (2017).
4. Naghashi, V., Dallago, G., Diallo, A. B. & Boukadoum, M. Univariate and multivariate time-series methods to forecast dairy income. In *2nd AAAI Workshop on AI for Agriculture and Food Systems* (2023).
5. Nie, Y., Nguyen, N. H., Sinthong, P. & Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with transformers. arXiv:2211.14730 (2022).
6. Wang, S. *et al.* Timemixer: Decomposable multiscale mixing for time series forecasting. arXiv:2405.14616 (2024).
7. Wang, Z. et al. Is mamba effective for time series forecasting?. *Neurocomputing* **619**, 129178 (2025).
8. Cho, K. *et al.* Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv:1406.1078 (2014).
9. Liseune, A., Salamone, M., Van den Poel, D., Van Ranst, B. & Hostens, M. Predicting the milk yield curve of dairy cows in the subsequent lactation period using deep learning. *Comput. Electron. Agric.* **180**, 105904 (2021).
10. Salamone, M. et al. Prediction of first test day milk yield using historical records in dairy cows. *Animal* **16**, 100658 (2022).
11. Contreras, J., Espinola, R., Nogales, F. J. & Conejo, A. J. Arima models to predict next-day electricity prices. *IEEE Trans. Power Syst.* **18**, 1014–1020 (2003).
12. Oreshkin, B. N., Carpov, D., Chapados, N. & Bengio, Y. N-beats: Neural basis expansion analysis for interpretable time series forecasting. arXiv:1905.10437 (2019).
13. Wu, N., Green, B., Ben, X. & O'Banion, S. Deep transformer models for time series forecasting: The influenza prevalence case. arXiv:2001.08317 (2020).
14. Zhou, H. et al. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 11106–11115 (2021).
15. Wu, H., Xu, J., Wang, J. & Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Adv. Neural Inf. Process. Syst.* **34**, 22419–22430 (2021).
16. Zhou, T. *et al.* Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning*, 27268–27286 (PMLR, 2022).
17. Zhang, Y. & Yan, J. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations* (2022).
18. Liu, Y. *et al.* itransformer: Inverted transformers are effective for time series forecasting. arXiv:2310.06625 (2023).
19. Salinas, D., Flunkert, V., Gasthaus, J. & Januschowski, T. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *Int. J. Forecast.* **36**, 1181–1191 (2020).
20. Zeng, A., Chen, M., Zhang, L. & Xu, Q. Are transformers effective for time series forecasting? In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 11121–11128 (2023).
21. Li, Z., Qi, S., Li, Y. & Xu, Z. Revisiting long-term time series forecasting: An investigation on linear mapping. arXiv:2305.10721 (2023).
22. Gu, A. & Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. arXiv:2312.00752 (2023).
23. Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural Comput.* **9**, 1735–1780 (1997).
24. Wang, C., Liu, Z., Wei, H., Chen, L. & Zhang, H. Hybrid deep learning model for short-term wind speed forecasting based on time series decomposition and gated recurrent unit. *Complex Syst. Model. Simul.* **1**, 308–321 (2021).
25. Lawi, A., Mesra, H. & Amir, S. Implementation of long short-term memory and gated recurrent units on grouped time-series data to predict stock prices accurately. *J. Big Data* **9**, 89 (2022).
26. Lin, S. *et al.* Segrnn: Segment recurrent neural network for long-term time series forecasting. arXiv:2308.11200 (2023).
27. Wang, S. *et al.* Timemixer++: A general time series pattern machine for universal predictive analysis. arXiv:2410.16032 (2024).
28. Wu, H. *et al.* Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations* (2022).
29. Paszke, A. *et al.* Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems* **32** (2019).
30. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. arXiv:1412.6980 (2014).

Author contributions

V.N. did Conceptualization, Methodology, Visualisation, Data curation, Writing- Original draft preparation. M.B. carried out Supervision, Writing- Reviewing and Editing. A.D. carried out Supervision, Investigation, Writing- Reviewing and Editing.

Funding

There is no funding to declare.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to A.B.D.

BIBLIOGRAPHIE

Alghamdi, T., Elgazzar, K., Bayoumi, M., Sharaf, T. et Shah, S. (2019). Forecasting traffic congestion using arima modeling. Dans *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, 1227–1232. IEEE.

Benedet, A., Manuelian, C., Zidi, A., Penasa, M. et De Marchi, M. (2019). Invited review : β-hydroxybutyrate concentration in blood and milk and its associations with cow performance. *Animal*, *13*(8), 1676–1689.

Bengio, Y., Simard, P. et Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, *5*(2), 157–166.

Bontempi, G., Ben Taieb, S. et Le Borgne, Y.-A. (2012). Machine learning strategies for time series forecasting. In *European Big Data Management and Analytics Summer School* 62–77. Springer.

Borchers, M., Chang, Y., Proudfoot, K., Wadsworth, B., Stone, A. et Bewley, J. (2017). Machine-learning-based calving prediction from activity, lying, and ruminating behaviors in dairy cattle. *Journal of dairy science*, *100*(7), 5664–5674.

Box, G. E., Jenkins, G. M., Reinsel, G. C. et Ljung, G. M. (2015). *Time series analysis : forecasting and control*. John Wiley & Sons.

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5–32.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A. *et al.* (2020). Language models are few-shot learners. *Advances in neural information processing systems*, *33*, 1877–1901.

Bruinsma, J. (2017). *World agriculture : towards 2015/2030 : an FAO study*. Routledge.

Bruinsma, J. *et al.* (2012). European and central asian agriculture towards 2030 and 2050. *Policy studies on rural transition*, (2012-1).

Challu, C., Olivares, K. G., Oreshkin, B. N., Ramirez, F. G., Canseco, M. M. et Dubrawski, A. (2023). Nhits : Neural hierarchical interpolation for time series forecasting. Dans *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 6989–6997.

Chen, P., Zhang, Y., Cheng, Y., Shu, Y., Wang, Y., Wen, Q., Yang, B. et Guo, C. (2024). Pathformer : Multi-scale transformers with adaptive pathways for time series forecasting. *arXiv e-prints*, arXiv–2402.

Chen, S.-A., Li, C.-L., Yoder, N., Arik, S. O. et Pfister, T. (2023a). Tsmixer : An all-mlp architecture for time series forecasting. *arXiv preprint arXiv :2303.06053*.

Chen, T. et Guestrin, C. (2016). Xgboost : A scalable tree boosting system. Dans *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794.

Chen, Z., Ma, M., Li, T., Wang, H. et Li, C. (2023b). Long sequence time-series forecasting with deep learning : A survey. *Information Fusion*, *97*, 101819.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. et Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv :1406.1078*.

Cirstea, R.-G., Guo, C., Yang, B., Kieu, T., Dong, X. et Pan, S. (2022). Triformer : Triangular, variable-specific attentions for long sequence multivariate time series forecasting–full version. *arXiv preprint arXiv :2204.13767*.

Contreras, J., Espinola, R., Nogales, F. J. et Conejo, A. J. (2003). Arima models to predict next-day electricity prices. *IEEE transactions on power systems*, *18*(3), 1014–1020.

Das, A., Kong, W., Leach, A., Mathur, S., Sen, R. et Yu, R. (2023). Long-term forecasting with tide : Time-series dense encoder. *arXiv preprint arXiv :2304.08424*.

Dematawewa, C., Pearson, R. et VanRaden, P. (2007). Modeling extended lactations of holsteins. *Journal of Dairy Science*, *90*(8), 3924–3936.

Dosovitskiy, A. (2020). An image is worth 16x16 words : Transformers for image recognition at scale. *arXiv preprint arXiv :2010.11929*.

Ehrlich, J. (2011). Quantifying shape of lactation curves, and benchmark curves for common dairy breeds and parities. *The Bovine Practitioner*, 88–95.

EL-TAHAWY, A. S. et EL-FAR, A. H. (2010). Influences of somatic cell count on milk composition and dairy farm profitability. *International journal of dairy technology*, *63*(3), 463–469.

Elsaraiti, M., Ali, G., Musbah, H., Merabet, A. et Little, T. (2021). Time series analysis of electricity consumption forecasting using arima model. Dans *2021 IEEE Green technologies conference (GreenTech)*, 259–262. IEEE.

Erickson, P. S. et Kalscheur, K. F. (2020). Nutrition and feeding of dairy cattle. In *Animal agriculture* 157–180. Elsevier.

Ferreira, M. A., Higdon, D. M., Lee, H. K. et West, M. (2006). Multi-scale and hidden resolution time series models. *Bayesian Analysis*.

Frasco, C. G., Radmacher, M., Lacroix, R., Cue, R., Valtchev, P., Robert, C., Boukadoum, M., Sirard, M.-A. et Diallo, A. B. (2020). Towards an effective decision-making system based on cow profitability using deep learning. Dans *ICAART (2)*, 949–958.

Gers, F. A., Schmidhuber, J. et Cummins, F. (2000). Learning to forget : Continual prediction with lstm. *Neural computation*, *12*(10), 2451–2471.

Goodfellow, I., Bengio, Y., Courville, A. et Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge.

Grzesiak, W., Lacroix, R., Wójcik, J. et Blaszczyk, P. (2003). A comparison of neural network and multiple regression predictions for 305-day lactation yield using partial lactation records. *Canadian Journal of Animal Science*, *83*(2), 307–310.

Gu, A. et Dao, T. (2023). Mamba : Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv :2312.00752*.

Guo, S., Lin, Y., Wan, H., Li, X. et Cong, G. (2021). Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting. *IEEE Transactions on Knowledge and Data Engineering*, *34*(11), 5415–5428.

Han, S., Mao, H. et Dally, W. J. (2015). Deep compression : Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv :1510.00149*.

Hewamalage, H., Bergmeir, C. et Bandara, K. (2021). Recurrent neural networks for time series forecasting : Current status and future directions. *International Journal of Forecasting*, *37*(1), 388–427.

Hochreiter, S. et Schmidhuber, J. (1997a). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Hochreiter, S. et Schmidhuber, J. (1997b). Lstm can solve hard long time lag problems. *Advances in neural information processing systems*, 473–479.

Hou, B.-J. et Zhou, Z.-H. (2020). Learning with interpretable structure from gated rnn. *IEEE transactions on neural networks and learning systems*, *31*(7), 2267–2279.

Hyndman, R. J. et Athanasopoulos, G. (2018). *Forecasting : principles and practice*. OTexts.

Hyndman, R. J. et Khandakar, Y. (2008). Automatic time series forecasting : the forecast package for r. *Journal of statistical software*, *27*, 1–22.

Hyndman, R. J. et Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International journal of forecasting*, *22*(4), 679–688.

Jiang, B., Wu, Q., Yin, X., Wu, D., Song, H. et He, D. (2019). Flyolov3 deep learning for key parts of dairy cow body detection. *Computers and Electronics in Agriculture*, *166*, 104982.

Jin, M., Wang, S., Ma, L., Chu, Z., Zhang, J. Y., Shi, X., Chen, P.-Y., Liang, Y., Li, Y.-F., Pan, S. *et al.* (2023). Time-llm : Time series forecasting by reprogramming large language models. *arXiv preprint arXiv :2310.01728*.

Kammerdiner, A., Xanthopoulos, P. et Pardalos, P. M. (2007). Numerical limitations in application of vector autoregressive modeling and granger causality to analysis of eeg time series. Dans *AIP Conference Proceedings*, volume 953, 232–245. American Institute of Physics.

Kim, T., Kim, J., Tae, Y., Park, C., Choi, J.-H. et Choo, J. (2021). Reversible instance normalization for accurate time-series forecasting against distribution shift. Dans *International Conference on Learning Representations*.

Kingma, D. P. et Ba, J. (2014). Adam : A method for stochastic optimization. *arXiv preprint arXiv :1412.6980*.

Kitaev, N., Kaiser, L. et Levskaya, A. (2019). Reformer : The efficient transformer. Dans *International Conference on Learning Representations*.

Kohn, R. (2007). Use of milk or blood urea nitrogen to identify feed management inefficiencies and

estimate nitrogen excretion by dairy cattle and other animals. Dans *Florida Ruminant Nutrition Symposium*, 30–31. Citeseer.

Lai, G., Chang, W.-C., Yang, Y. et Liu, H. (2018). Modeling long-and short-term temporal patterns with deep neural networks. Dans *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 95–104.

Lawi, A., Mesra, H. et Amir, S. (2022). Implementation of long short-term memory and gated recurrent units on grouped time-series data to predict stock prices accurately. *Journal of Big Data*, *9*(1), 89.

LeCun, Y., Bengio, Y. et Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436–444.

LeCun, Y., Bottou, L., Bengio, Y. et Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X. et Yan, X. (2019). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems*, *32*.

Li, Z., Qi, S., Li, Y. et Xu, Z. (2023a). Revisiting long-term time series forecasting : An investigation on linear mapping. *arXiv preprint arXiv :2305.10721*.

Li, Z. L., Zhang, G. W., Yu, J. et Xu, L. Y. (2023b). Dynamic graph structure learning for multivariate time series forecasting. *Pattern Recognition*, *138*, 109423.

Lim, B. et Zohren, S. (2021). Time-series forecasting with deep learning : a survey. *Philosophical Transactions of the Royal Society A*, *379*(2194), 20200209.

Lim, D.-H., Mayakrishnan, V., Lee, H.-J., Ki, K.-S., Kim, T.-I. et Kim, Y. (2020). A comparative study on milk composition of jersey and holstein dairy cows during the early lactation. *Journal of Animal Science and Technology*, *62*(4), 565.

Lin, S., Lin, W., Wu, W., Zhao, F., Mo, R. et Zhang, H. (2023). Segrnn : Segment recurrent neural network for long-term time series forecasting. *arXiv preprint arXiv :2308.11200*.

Lipton, Z. C., Berkowitz, J. et Elkan, C. (2015). A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv :1506.00019*.

Liseune, A., Salamone, M., Van den Poel, D., Van Ranst, B. et Hostens, M. (2020). Leveraging latent representations for milk yield prediction and interpolation using deep learning. *Computers and Electronics in Agriculture*, *175*, 105600.

Liseune, A., Salamone, M., Van den Poel, D., Van Ranst, B. et Hostens, M. (2021). Predicting the milk yield curve of dairy cows in the subsequent lactation period using deep learning. *Computers and Electronics in Agriculture*, *180*, 105904.

Liu, C., Yang, S., Xu, Q., Li, Z., Long, C., Li, Z. et Zhao, R. (2024). Spatial-temporal large language model for traffic prediction. *arXiv preprint arXiv :2401.10134*.

Liu, M., Ren, S., Ma, S., Jiao, J., Chen, Y., Wang, Z. et Song, W. (2021a). Gated transformer networks for multivariate time series classification. *arXiv preprint arXiv :2103.14438*.

Liu, M., Zeng, A., Chen, M., Xu, Z., Lai, Q., Ma, L. et Xu, Q. (2022). Scinet : Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, *35*, 5816–5828.

Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A. X. et Dustdar, S. (2021b). Pyraformer : Low-complexity pyramidal attention for long-range time series modeling and forecasting. Dans *International conference on learning representations*.

Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L. et Long, M. (2023). itransformer : Inverted transformers are effective for time series forecasting. Dans *The Twelfth International Conference on Learning Representations*.

Lütkepohl, H. (2005). *New introduction to multiple time series analysis*. Springer Science & Business Media.

Ma, N., Pan, L., Chen, S. et Liu, B. (2020). Nb-iot estrus detection system of dairy cows based on lstm networks. Dans *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, 1–5. IEEE.

Makridakis, S., Spiliotis, E. et Assimakopoulos, V. (2018). Statistical and machine learning forecasting methods : Concerns and ways forward. *PloS one*, *13*(3), e0194889.

Martínez, F., Charte, F., Frías, M. P. et Martínez-Rodríguez, A. M. (2022). Strategies for time series forecasting with generalized regression neural networks. *Neurocomputing*, *491*, 509–521.

Miao, H., Liu, Z., Zhao, Y., Guo, C., Yang, B., Zheng, K. et Jensen, C. S. (2024). Less is more : Efficient time series dataset condensation via two-fold modal matching–extended version. *arXiv preprint arXiv :2410.20905*.

Naghashi, V., Boukadoum, M. et Diallo, A. B. (2025a). A multiscale model for multivariate time series forecasting. *Scientific Reports*, *15*(1), 1565.

Naghashi, V., Boukadoum, M. et Diallo, A. B. (2025b). Should we reconsider rnns for time-series forecasting ? *AI*, *6*(5), 90.

Naghashi, V., Dallago, G., Diallo, A. B. et Boukadoum, M. (2023). Univariate and multivariate time-series methods to forecast dairy income. Dans *2nd AAAI Workshop on AI for Agriculture and Food Systems*.

Nie, Y., Nguyen, N. H., Sinthong, P. et Kalagnanam, J. (2022). A time series is worth 64 words : Long-term forecasting with transformers. *arXiv preprint arXiv :2211.14730*.

Oreshkin, B. N., Carpov, D., Chapados, N. et Bengio, Y. (2019). N-beats : Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv :1905.10437*.

Oreshkin, B. N., Element, A., Carpov, D., Chapados, N. et Bengio, Y. N-beats : Neural basis expansion

analysis for interpretable time series forecasting.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L. *et al*. (2019). Pytorch : An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, *32*.

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, *1*, 81–106.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I. *et al*. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, *1*(8), 9.

Rahnemoonfar, M. et Sheppard, C. (2017). Deep count : fruit counting based on deep simulated learning. *Sensors*, *17*(4), 905.

Ruder, S., Peters, M. E., Swayamdipta, S. et Wolf, T. (2019). Transfer learning in natural language processing. Dans *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics : Tutorials*, 15–18.

Salamone, M., Adriaens, I., Vervaet, A., Opsomer, G., Atashi, H., Fievez, V., Aernouts, B. et Hostens, M. (2022). Prediction of first test day milk yield using historical records in dairy cows. *animal*, *16*(11), 100658.

Salinas, D., Flunkert, V., Gasthaus, J. et Januschowski, T. (2020). Deepar : Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, *36*(3), 1181–1191.

Schaeffer, L. et Jamrozik, J. (1996). Multiple-trait prediction of lactation yields for dairy cows. *Journal of Dairy Science*, *79*(11), 2044–2055.

Shabani, A., Abdi, A., Meng, L. et Sylvain, T. (2022). Scaleformer : iterative multi-scale refining transformers for time series forecasting. *arXiv preprint arXiv :2206.04038*.

Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D. et Stefanovic, D. (2016). Deep neural networks based recognition of plant diseases by leaf image classification. *Computational intelligence and neuroscience*, *2016*.

Song, X., Zhang, G., Liu, F., Li, D., Zhao, Y. et Yang, J. (2016). Modeling spatio-temporal distribution of soil moisture by deep learning-based cellular automata model. *Journal of Arid Land*, *8*(5), 734–748.

Strubell, E., Ganesh, A. et McCallum, A. (2019). Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv :1906.02243*.

Sun, C., Shrivastava, A., Singh, S. et Gupta, A. (2017). Revisiting unreasonable effectiveness of data in deep learning era. *Proceedings of the IEEE international conference on computer vision*, 843–852.

Sun, Z., Di, L., Fang, H. et Burgess, A. (2020). Deep learning classification for crop types in north dakota. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *13*, 2200–2213.

Thomas, R. M., Bruin, W., Zhutovsky, P. et van Wingen, G. (2020). Dealing with missing data, small sample sizes, and heterogeneity in machine learning studies of brain disorders. In *Machine learning*

249–266. Elsevier.

Ushikubo, S., Kubota, C. et Ohwada, H. (2017). The early detection of subclinical ketosis in dairy cows using machine learning methods. Dans *Proceedings of the 9th International Conference on Machine Learning and Computing*, 38–42.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. et Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Vijayakumar, M., Park, J. H., Ki, K. S., Lim, D. H., Kim, S. B., Park, S. M., Jeong, H. Y., Park, B. Y. et Kim, T. I. (2017). The effect of lactation number, stage, length, and milking frequency on milk yield in korean holstein dairy cows using automatic milking system. *Asian-Australasian journal of animal sciences*, *30*(8), 1093.

Wang, C., Liu, Z., Wei, H., Chen, L. et Zhang, H. (2021a). Hybrid deep learning model for short-term wind speed forecasting based on time series decomposition and gated recurrent unit. *Complex System Modeling and Simulation*, *1*(4), 308–321.

Wang, S., Li, J., Shi, X., Ye, Z., Mo, B., Lin, W., Ju, S., Chu, Z. et Jin, M. (2024a). Timemixer++ : A general time series pattern machine for universal predictive analysis. *arXiv preprint arXiv :2410.16032*.

Wang, S., Wu, H., Shi, X., Hu, T., Luo, H., Ma, L., Zhang, J. Y. et Zhou, J. (2024b). Timemixer : Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv :2405.14616*.

Wang, S., Zhang, M., Miao, H., Peng, Z. et Yu, P. S. (2022). Multivariate correlation-aware spatio-temporal graph convolutional networks for multi-scale traffic prediction. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *13*(3), 1–22.

Wang, S., Zhang, M., Miao, H. et Yu, P. S. (2021b). Mt-stnets : Multi-task spatial-temporal networks for multi-scale traffic prediction. Dans *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, 504–512. SIAM.

Wang, Z., Kong, F., Feng, S., Wang, M., Yang, X., Zhao, H., Wang, D. et Zhang, Y. (2025). Is mamba effective for time series forecasting ? *Neurocomputing*, *619*, 129178.

Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J. et Long, M. (2022). Timesnet : Temporal 2d-variation modeling for general time series analysis. Dans *The eleventh international conference on learning representations*.

Wu, H., Xu, J., Wang, J. et Long, M. (2021). Autoformer : Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, *34*, 22419–22430.

Wu, N., Green, B., Ben, X. et O'Banion, S. (2020a). Deep transformer models for time series forecasting : The influenza prevalence case. *arXiv preprint arXiv :2001.08317*.

Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X. et Zhang, C. (2020b). Connecting the dots : Multivariate time series forecasting with graph neural networks. Dans *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 753–763.

Xu, J., Zhu, Y., Zhong, R., Lin, Z., Xu, J., Jiang, H., Huang, J., Li, H. et Lin, T. (2020). Deepcropmapping : A multi-temporal deep learning approach with improved spatial generalizability for dynamic corn and soybean mapping. *Remote Sensing of Environment*, *247*, 111946.

Yang, Q., Liu, Y., Chen, T. et Tong, Y. (2019). Federated machine learning : Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *10*(2), 1–19.

Zeng, A., Chen, M., Zhang, L. et Xu, Q. (2023). Are transformers effective for time series forecasting ? Dans *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 11121–11128.

Zhang, G., Patuwo, B. E. et Hu, M. Y. (1998). Forecasting with artificial neural networks : : The state of the art. *International journal of forecasting*, *14*(1), 35–62.

Zhang, G. P. (2003). Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, *50*, 159–175.

Zhang, J., Zhu, Y., Zhang, X., Ye, M. et Yang, J. (2018). Developing a long short-term memory (lstm) based model for predicting water table depth in agricultural areas. *Journal of hydrology*, *561*, 918–929.

Zhang, Y. et Yan, J. (2022). Crossformer : Transformer utilizing cross-dimension dependency for multivariate time series forecasting. Dans *The eleventh international conference on learning representations*.

Zhang, Y. et Yan, J. (2023). Crossformer : Transformer utilizing cross-dimension dependency for multivariate time series forecasting. Dans *The eleventh international conference on learning representations*.

Zhao, K., Guo, C., Cheng, Y., Han, P., Zhang, M. et Yang, B. (2023). Multiple time series forecasting with dynamic graph modeling. *Proceedings of the VLDB Endowment*, *17*(4), 753–765.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H. et Zhang, W. (2021). Informer : Beyond efficient transformer for long sequence time-series forecasting. Dans *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 11106–11115.

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L. et Jin, R. (2022). Fedformer : Frequency enhanced decomposed transformer for long-term series forecasting. Dans *International conference on machine learning*, 27268–27286. PMLR.

Zhou, T., Niu, P., Sun, L., Jin, R. *et al.* (2023). One fits all : Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, *36*, 43322–43355.

Zin, T. T., Phyo, C. N., Tin, P., Hama, H. et Kobayashi, I. (2018). Image technology based cow identification system using deep learning. Dans *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, 236–247.

Zivot, E. et Wang, J. (2006). Vector autoregressive models for multivariate time series. *Modeling Financial Time Series with S-Plus®*, 385–429.