UNIVERSITÉ DU QUÉBEC À MONTRÉAL

ESTIMATION D'INDICES SUR L'ÉTAT DE SANTÉ MENTALE PAR ANALYSE AUTOMATISÉE DE PRODUCTIONS TEXTUELLES

THÈSE

PRÉSENTÉE

COMME EXIGENCE PARTIELLE

DU DOCTORAT EN INFORMATIQUE

PAR

DIEGO MAUPOMÉ

MARS 2024

UNIVERSITÉ DU QUÉBEC À MONTRÉAL

GATHERING SIGNS OF MENTAL HEALTH CONCERNS THROUGH THE AUTOMATED ANALYSIS OF TEXTUAL PRODUCTION

THESIS

PRESENTED

AS PARTIAL REQUIREMENT

TO THE PH.D IN COMPUTER SCIENCE

BY

DIEGO MAUPOMÉ

MARCH 2024

UNIVERSITÉ DU QUÉBEC À MONTRÉAL
Service des bibliothèques

Avertissement

La diffusion de cette thèse se fait dans le respect des droits de son auteur, qui a signé le formulaire *Autorisation de reproduire et de diffuser un travail de recherche de cycles supérieurs* (SDU-522 – Rév.12-2023). Cette autorisation stipule que «conformément à l'article 11 du Règlement no 8 des études de cycles supérieurs, [l'auteur] concède à l'Université du Québec à Montréal une licence non exclusive d'utilisation et de publication de la totalité ou d'une partie importante de [son] travail de recherche pour des fins pédagogiques et non commerciales. Plus précisément, [l'auteur] autorise l'Université du Québec à Montréal à reproduire, diffuser, prêter, distribuer ou vendre des copies de [son] travail de recherche à des fins non commerciales sur quelque support que ce soit, y compris l'Internet. Cette licence et cette autorisation n'entraînent pas une renonciation de [la] part [de l'auteur] à [ses] droits moraux ni à [ses] droits de propriété intellectuelle. Sauf entente contraire, [l'auteur] conserve la liberté de diffuser et de commercialiser ou non ce travail dont [il] possède un exemplaire.»

CONTENTS

LIST OF TABLES

LIST OF FIGURES

RÉSUMÉ

Les troubles et souffrances psychiques peuvent avoir des conséquences graves, allant de l'incapacité à la mort (World Health Organization, 2013; Canadian Mental Health Association, 2020). Un enjeu notoire autour de leur dépistage et dans le déploiement d'efforts d'intervention auprès des personnes concernées, est la difficulté d'accès à des ressources de soutien et à des soins adéquats. Cette situation est exacerbée par la stigmatisation des troubles de santé mentale (Henderson et al., 2013). Or, les personnes à risque font souvent part de leurs vécus et inquiétudes sur des plateformes numériques de plus en plus variées et de plus en plus utilisées. Ainsi, il existe un intérêt croissant dans l'analyse automatique de ces informations afin de rassembler des indices sur l'état mental de personnes potentiellement à risque (De Choudhury et al., 2013; Merchant et al., 2019).

La présente thèse examine l'analyse automatisée de contenus textuels partagés en ligne à des fins d'évaluation de risques liés à la santé mentale. Spécifiquement, le travail présenté concerne l'utilisation de méthodes d'apprentissage automatique pour produire des analyses procédurales de l'état de santé mentale d'une personne à partir de ses écrits partagés en ligne. Ces efforts comportent un bilan des défis particuliers d'une telle entreprise du point de vue de l'apprentissage automatique. Un premier obstacle est le manque de données. Les algorithmes d'apprentissage automatique nécessitent une compilation d'exemples à partir desquels des inférences seront effectuées. Les approches fines, capables d'extraire des structures complexes, ont besoin d'un grand volume de données pour atteindre de bonnes performances dans leurs capacités de généralisation. Or, la collecte de données en matière de santé mentale est coûteuse en raison de l'expertise nécessaire, rendant difficile la production de ressources vastes. Un deuxième obstacle, plus fondamental, est le balisage approprié des inférences à effectuer. Bien que les méthodes d'apprentissage automatique puissent faire des inférences à partir des données, elles doivent le faire à partir de certains choix de structures qui leur sont fournis, *i.e.* des hypothèses d'apprentissage. Une vaste littérature sur l'analyse de données textuelles peut aider à guider ces choix. Toutefois, son application à des fins d'évaluation de l'état de santé mentale est particulière dans la mesure où les signaux à sonder ont un lien indirect avec le texte et peuvent être diffus à travers une grande quantité d'information. Un troisième obstacle, plus technique, est le fait que l'analyse détaillée de textes peut nécessiter un coût calculatoire qui ne passe pas à l'échelle imposée par les grandes quantités de texte que les personnes peuvent produire sur les plateformes numériques.

La présente thèse comporte ainsi une série de contributions visant à aborder ces obstacles. Deux de ces contributions se concentrent sur l'élaboration d'approches efficaces pour la modélisation de la langue, tant en termes de nombre de paramètres nécessaires qu'en termes de temps de calcul. Ces travaux s'inscrivent dans un objectif de diminution des besoins en ressources de calcul des algorithmes de modélisation de la langue. Deux autres contributions étudient le déploiement de ce type d'approches pour l'analyse de l'état de santé mentale à travers la langue écrite, en particulier dans cas différentes souffrances psychiques. Ces efforts s'articulent autour de considérations pratiques particulières comme le nombre limité de données et le besoin de prédictions dans des délais opportuns. Enfin, une contribution supplémentaire se penche sur la transposition d'éléments de modélisation entre différentes souffrances psychiques.

ABSTRACT

Threats to mental health are highly prevalent and can have grave consequences including disability and death (World Health Organization, 2013; Canadian Mental Health Association, 2020). A major issue in addressing these threats is lack of access to care, which is often exacerbated by the stigma surrounding mental illness (Henderson et al., 2013). At-risk persons will often share their experiences and concerns on an ever-increasing variety of online platforms. Consequently, there has been increased research interest in the analysis of this information to gain insights into the mental status of at-risk persons. (De Choudhury et al., 2013; Merchant et al., 2019).

The present thesis examines the automated analysis of textual content for mental health assessment. Specifically, the work described investigates the use of machine learning to infer procedures mapping written language content to an assessment of the mental health of its author. In particular, it provides an account of the distinct and important challenges of this endeavour from a machine learning perspective. A primary challenge is the scarcity of data from which to make inferences. Indeed, machine learning algorithms require curated examples to make inferences. Further, sophisticated approaches which can capture complex relationships in the data require large amounts of data to avoid collapse into finely tuned but generally inadequate solutions. The need for significant clinical expertise makes data expensive and therefore scarce. A more fundamental challenge has to do with the delineation of inferences. While machine learning algorithms can detect patterns in data, they need to be provided definite structure on which to base these patterns. To be sure, a vast corpus of research into the analysis of natural language data provides some insight. Nevertheless, its application to the problem at hand gives rise to a peculiar setting in that the signals to detect—those pertaining to a given aspect of mental health—may be dispersed throughout large amounts of text and have an oblique relationship to it. A third, more technical challenge is the computational cost of analysis. Examining written language in great detail may require intensive computation, which is incompatible with the large quantities of text that may be comprised in a single examination.

This thesis comprises a series of contributions addressing these challenges. Two of these focus on the development of parameter- and time-efficient approaches to modelling written language. The work they comprise aims to understand how approaches to modelling natural language data can be made computationally efficient. Two other contributions study their application towards evaluating risk to mental health. These efforts contend with several practical considerations such as limited data availability and the need for time-sensitive decisions. A final contribution studies how portable the more abstract components of inferences can be between different behavioural concerns.

INTRODUCTION

Mental illness accounts for 13% of the total global burden of disease. People suffering from mental health disorders have disproportionately higher rates of disability and mortality, with a risk of premature death up to 40 to 60% higher than that of the general population (World Health Organization, 2013). In Canada, 20% of the population will personally experience a mental health issue at any given year (Canadian Mental Health Association, 2020). Despite the high prevalence of mental illness, those affected and those at risk will struggle finding adequate help. In 2018, 76% of Canadians aged 15-34 with mental health concerns affirmed not having received care outside their family and friends (Statistics Canada, 2020).

Thus, there appears to exist widespread for mental health care together with a disconnect between care services and at-risk persons. These circumstances have given rise to a search for new avenues for the delivery of mental health care. This includes new channels for gathering signs of mental health concerns (McGorry & Mei, 2018; Ernala et al., 2019). Concurrently, the rising popularity of online discussion and networking platforms has created a vast increase in the availability of written language authored by affected and at-risk persons. These contents can offer insight into their outlooks and experiences. Consequently, recent years have been marked by a growth in research interest in developing automated means of parsing online text in the service of mental health.

Unfortunately, our understanding of language in general and how it relates to mental health in particular, while considerable, does not offer computable theories that could be leveraged in this direction. That is, there are no verified procedures for understanding written language or, indeed, for the detection of threats to mental health from it. In the absence of such procedural means, it is possible to employ machine learning. Machine learning algorithms are powerful means of inferring such computations. There are a great many things that humans know how to do without having an explanation of their process that is precise enough that it could be implemented as a computer program. For example, many people may be able, given a photograph of a person, to determine whether the portrayed subject is smiling or frowning. The explanation they might offer for their reasoning will typically operate on terms that are themselves difficult to describe or define, *e.g.* mouth, eyes. While seemingly mundane, some of these terms refer to concepts that are highly complex when considered in the simple language of computers and procedures. In such cases, one can attempt to derive those procedures automatically through machine learning. Put simply, from examples, machine learning techniques can produce some computation that reflects the examples provided. In the case of mental health assessment from text, an example could be a set of writings matched to an assessment of the mental statuses of their author. Given a number of these examples, machine learning techniques could infer how to arrive at these assessments from the sets of writings.

Nevertheless, this process can encounter a few obstacles. Firstly, for these algorithms to infer a viable procedure, one must provide them with a set of options. Indeed, these algorithms cannot learn without assumptions. In some cases, finding the correct assumptions may be as difficult a process as manually deriving a procedure. Fortunately, a large body of work has been devoted to the study of machine learning and human language. Further, one can also rely on previous research incursions into automated mental health assessment from text. Secondly, the inference algorithms charged with deriving the desired procedure will tend to take the path of least resistance, so to speak. That is, they will often find solutions that will be true to some of the examples provided, but not the phenomenon at large, making them of little practical use. Although this is less likely to occur when the assumptions provided or embedded in the algorithm are true to the underlying phenomenon, as mentioned, those may be difficult to derive. Alternatively, presenting the algorithms with more examples will make it more difficult for them to produce a less general solution, as a larger portion of the underlying phenomenon will be exposed. However, examples may be costly to produce. This is the case of examples that require human intervention to be produced. This cost only increases with the expertise required for the task. In the case of mental health assessment, producing example assessments of some aspect of mental health requires significant expertise, making examples considerably costly. Thirdly, even in settings where sound assumptions and sufficient data are available, there may be other considerations at play. Viable options may be theoretically possible but made impracticable by a high computational cost. Complex approaches—ones that encompass vast amounts of information or apply heavy processing to this information—though usually more discerning, will incur high computational cost.

All three of these challenges arise in attempting to assess the mental health of individuals from their textual productions. Thus, the work presented in this thesis seeks **to study computationally efficient machine learning approaches toward the representation of written natural language aimed at the application on mental health analysis**. Chapter 1 introduces prerequisite notions of machine learning and natural language processing in order to provide the theoretical and technical context of this investigation. These notions provide the building blocks necessary to understand the considerations in applying natural language processing to mental health assessment in Chapter 2. The subsequent chapters present the work put forth in this thesis. In Chapter 3, data scarcity is addressed by the use of authorship modelling as a means to extract information from text in order to predict depression symptoms. While experiments show these approaches to have predictive capabilities, they suffer from limitations in the patterns they are able to capture in text. Namely, they rely on counting the number of occurrences of words, foregoing some of the structure of written language. To remediate this, Chapter 4 presents more theoretically oriented work proposing ameliorations to some of the methods used in the preceding chapter in order to account for more complex aspects of written language while remaining efficient in terms of computational resources. In the same vein, Chapter 5 further explores these considerations by exploring how to model context in the representation of natural language.

Chapter 6 recounts a participation in a mental health risk assessment shared task. Several challenges around behavioural health assessment from written language are tackled using the approaches put forth in previous chapters. Finally, Chapter 7 steps slightly back from mental health risk assessment to delve into considerations around the nature of online mental health discourse. In so doing, it asks how specific to particular mental health language resources for machine learning should be.

This thesis presents five published articles:

- Chapter 3: **Diego Maupomé**, Maxime D. Armstrong, Fanny Rancourt, Marie-Jean Meurs (2021). Leveraging Textual Similarity to Predict Beck Depression Inventory Answers. *Proceedings of the* 34^{th} *Canadian Conference on Artificial Intelligence, CAI2021. (pp.1-12)*. Publisher: Canadian Artificial Intelligence Association.

- Chapter 4: **Diego Maupomé**, Fanny Rancourt, Maxime D. Armstrong, Marie-Jean Meurs (2021). Position Encoding Schemes for Linear Aggregation of Word Sequences. *Proceedings of the* 34^{th} *Canadian Conference on Artificial Intelligence, CAI2021. (pp.1-10)*. Publisher: Canadian Artificial Intelligence Association.

- Chapter 5: **Diego Maupomé**, Marie-Jean Meurs (2022). Contextualizer: Connecting the Dots of Context with Second-Order Attention. *Information* 2022, 13(6):290

- Chapter 6: **Diego Maupomé**, Maxime D. Armstrong, Fanny Rancourt, Thomas Soulas, Marie-Jean Meurs (2021). Early Detection of Signs of Pathological Gambling, Self-Harm and Depression through Topic Extraction and Neural Networks. *CLEF 2021 Conference and Labs of the Evaluation Forum. eRisk Shared Task (pp. 1031–1045)*

- Chapter 7 **Diego Maupomé**, Fanny Rancourt, Raouf Belbahar, Marie-Jean Meurs (2024). A Pretrained Language Model for Mental Health Risk Detection. *Proceedings of Machine Learning for Cognitive and Mental Health Workshop. AAAI 2024. (pp.23–28)*

In addition to the above contributions, my doctoral research work (2019–2022) has also led to the following contributions:

Peer reviewed contributions

- Book Chapter

  Diego Maupomé, Maxime D. Armstrong, Raouf Belbahar, Josselin Alezot, Rhon Balassiano, Fanny Rancourt, Marc Queudot, Sébastien Mosser, Marie-Jean Meurs (2022). Automatically Estimating the Severity of Multiple Symptoms Associated with Depression. *Early Detection of Mental Health Disorders by Social Media Monitoring (pp. 247-261)*, Publisher: Springer.

- Article

  Fanny Rancourt, **Diego Maupomé**, Marie-Jean Meurs (2022). On the Influence of Annotation Quality in Suicidal Risk Assessment from Text. *Proceedings of the* 35^{th} *Canadian Conference on Artificial Intelligence, CAI2022. (pp. 1-6).* Publisher: Canadian Artificial Intelligence Association.

- Article

  Seyed Habib Hosseini Saravani, Lancelot Normand, **Diego Maupomé**, Fanny Rancourt, Thomas Soulas, Sara Besharati, Anaelle Normand, Sébastien Mosser, Marie-Jean Meurs (2022). Measuring the severity of the signs of eating disorders using similarity-based models. *CLEF 2022 Conference and Labs of the Evaluation Forum. eRisk Shared Task (pp. 936–946)*

- Article

  Maxime D. Armstrong, **Diego Maupomé**, Meurs, Marie-Jean (2021). Topic Modeling in

Embedding Spaces for Depression Assessment. *Proceedings of the* 34^{th} *Canadian Conference on Artificial Intelligence, CAI2022. (pp. 1-7).* Publisher: Canadian Artificial Intelligence Association.

- Article

  Maxime D. Armstrong, **Diego Maupomé**, Meurs, Marie-Jean (2021). Topic Models for Assessment of Mental Health Issues. *Proceedings of the* 34^{th} *Canadian Conference on Artificial Intelligence, CAI2022. (pp. 1-6).* Publisher: Canadian Artificial Intelligence Association.

- Article

  Diego Maupomé, Marie-Jean Meurs (2020). Language Modeling with a General Second-Order RNN. *Proceedings of the 12th Language Resources and Evaluation Conference, LREC 2020 (pp. 4749-4753).*

Pre-prints

- **Diego Maupomé**, Fanny Rancourt, Thomas Soulas, Alexandre Lachance, Marie-Jean Meurs, Desislava Aleksandrova, Olivier Brochu Dufour, Igor Pontes, Rémi Cardon, Michel Simard, Sowmya Vajjala (2022). Automatic Text Simplification of News Articles in the Context of Public Broadcasting. *arXiv preprint arXiv:2212.13317 (pp. 1-10)*

- **Diego Maupomé**, Marie-Jean Meurs (2021). An Iterative Contextualization Algorithm with Second-Order Attention. *arXiv preprint arXiv:2103.02190 (pp. 1-10)*

CHAPTER 1
PRELIMINARY NOTIONS

This chapter introduces prerequisite notions of machine learning and natural language processing. First, Section 1.1 presents the basics of machine learning leading up to Section 1.2, which will go over specifics of neural networks. Finally, Section 1.3 veers into issues and modern practices in applying machine learning to natural language.

1.1 Machine learning basics

Broadly construed, the field of **Artificial Intelligence (AI)** seeks to study and develop machine-operable means to intelligent or rational decision-making. In turn, **machine learning** is the field of AI concerned with the automated derivation of computable approaches to set problems. That is, machine learning algorithms attempt to infer computable functions (algorithms) to address a given problem. This inference is based on curated data or experience. In the latter case, a so-called agent will manoeuvre in an environment and attempt to improve its actions based on the feedback from the environment. This is often termed **reinforcement learning** and will not be discussed in this thesis. In the case of learning from data, the process is static: although the algorithm may process a data point in some iterative manner, there is no proper interaction insofar as the algorithm cannot modify the data or produce new data points. The data remain as they are and the algorithm must make inferences from them. The data are also divided into singular examples, named **observations** or instances. An observation can be an image, a video segment or a single video frame, as well as a paragraph of text, a complete sentence or even a phrase. The demarcation of observations—what constitutes an observation—is set *a priori*, not inferred by the algorithm.

Further, the components of each observation may be separated into input and output. In such a case, the goal would be to infer a function which will approximate the output—also named **target**—to a satisfactory degree of exactitude given only the associated input. This type of setting is termed **supervised learning**. For example, a film review may be used as input, and its associated numerical score, as output. Then, the function to infer would be one which can produce the true numerical score when given the text of the review. In contrast, in **unsupervised learning**, no particular component of observations serves as a target. Consequently, unsupervised learning encompasses various types of inferences. Broadly speaking, nonetheless, they serve one or both of two purposes: to provide insights into the data in order to aid human analyses, or to help provide some appropriate **representation** of the data, for example, to serve another downstream machine learning problem. For example, in **clustering**, the aim is to group observations together based on some given measure of distance. In **feature selection**, the aim is to discard irrelevant or redundant attributes of the data, usually in order to facilitate further analyses. Of course, unsupervised learning can also be performed on data where observations otherwise include a target component but is disregarded. In both supervised and unsupervised learning learning, the type of mapping the desired function is expected to compute is called a **task**. That is, the goal is to infer a function that accomplishes the selected task (to some satisfactory degree) given the data.

This division—supervised v. unsupervised—is oft put forth as the chief distinction between machine learning algorithms (or areas of research at large) with all other aspects being subordinate to it. This is because it is a pragmatic distinction relating to what aspects of the data are available and what the practitioner wishes to accomplish. Nonetheless, there is no singular taxonomy of machine learning algorithms with some categorizations being hierarchically antecedent to others. In fact, many of these categorizations are largely independent of each other. As such, it is perhaps best to think of these categorizations as important aspects of machine learning algorithms, and of the associated categories, as contrasting options with regard to those aspects. Among these aspects are:

Supervised v. Unsupervised In the case of supervised learning, each observation will have an expected output associated with the input. This **annotation** may be of fixed format, as in analysis tasks (classification and regression) or of variable format, as in machine translation. In unsupervised learning, there is no given output for the observations in the dataset. Unsupervised learning may be a goal unto itself to aid human analysis of the data, or it may be done as an ancillary task to help some other downstream objective. When this objective is supervised learning, the combination is termed **semi-supervised learning**. In some cases of unsupervised learning, the function will be expected to produce portions of the input given other parts of it. This is sometimes termed **self-supervised learning**.

Classification v. Regression In supervised learning, the outputs may be of fixed format. If these outputs are nominal or ordinal in nature, this is termed **classification** because the function is expected to *classify* the observation, *i.e.* determine which of the possible classes it belongs to.

In classification, the targets are known as labels. When the targets are continuous, whether on an interval or ratio scale, the task is termed a **regression** task.

Binary v. Categorical The special case of classification where only two classes exist is termed binary classification. On the contrary, categorical classification involves more than two classes. By default, in either case, these classes are assumed to be mutually exclusive. In contrast, the term multi-label classification designates a setting where observations may belong to multiple classes.

Parametric v. Non-Parametric Often the learning of a task involves the adjustment of a set of objects (usually numbers) characterizing the function being learned. These objects are termed parameters. If the set of parameters is of fixed size, the approach is termed parametric. If, however, the number of parameters is unlimited and new parameters may be added as part of the learning procedure, the approach is termed non-parametric.

1.1.1 Optimization

The many different aspects of machine learning algorithms notwithstanding, all machine learning is rooted in optimization. Simply stated, we wish to find the function that best fits the data at our disposal. Of course, this requires a quantifiable criterion describing how satisfactory this so-called fit is. This criterion is termed a **loss function**, as it acts as the cost of misalignment with respect to the data in this task. This loss function, $\mathcal{L}$ will assign a loss value to any particular choice of function, f, when applied to the data, D, $\mathcal{L}(f, D)$. **Training** is then finding a function that suitably minimizes the loss:

$$\min_f \mathcal{L}(f, D)$$

This is perhaps most intuitive in a supervised setting, where each observation contains an expected output—or target—that the function attempts to match. The loss function can then be some measure of how incorrect a set of predicted values are with respect to the true targets. A straightforward choice would be the proportion of observations for which the function produced the wrong output: the **zero-one function**. Given a dataset $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^N$, $N \in \mathbb{N}$, where $x^{(i)}$ is the input and $y^{(i)}$, the target of the i-th observation, $i = 1, \ldots, N$, the loss incurred by f would be computed as:

$$\mathcal{L}_{0|1}(f, D) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{f(x^{(i)}) \neq y^{(i)}}$$

However, this particular choice of loss function may not be suited to all contexts. Arguably, it is not very informative, at least on a per-observation basis, as it takes on a very limited range of values (0 or 1). It will only reflect whether a prediction, $f(x_{(i)})$, is equal to the target, with no measure of degree.

Indeed, an appropriate choice of loss function will be contingent on the type of data at play as well as other considerations concerning the function to retrieve and the **training** procedure. As an example, in regression, interest will lie not in whether there is a difference between the predictions and the targets but in its magnitude. Consequently, it may be interesting to minimize the average absolute difference between prediction and target. Averaging across the dataset yields the **Mean Absolute Error (MAE)**:

$$\mathrm{MAE}(f, D) = \frac{1}{N}\sum_{i=1}^{N} |y^{(i)} - f(x^{(i)})|,$$

assuming $y^{(i)}, f(x^{(i)}) \in \mathbb{R}$, $i = 1, \ldots, N$.

A common alternative, the **Mean Square Error (MSE)**, sets the loss incurred by each prediction as proportional to the square of its deviation from the expected output:

$$\mathrm{MSE}(f, D) = \frac{1}{N}\sum_{i=1}^{N} |y^{(i)} - f(x^{(i)})|^2$$

Both loss functions can be used in similar settings. However, the quadratic growth of MSE with respect to the deviation from the target will make it much more sensitive to large discrepancies in prediction. The choice between them will vary depending on the specifics of the problem setting. This is even more so the case in unsupervised learning, where there is a wide range of desiderata in the transformation that the function must perform.

In any case, the loss function will serve to compare different functions in order to find one such function yielding a satisfactory loss value. A crucial consideration is that this particular choice will be made from a definite set of options. That is, no training procedure can select the best prediction function among all conceivable functions. Therefore, the practitioner must put forth a *family* of functions, $\mathcal{F}$. From this set, the training procedure will attempt to find a function that suitably minimizes the loss:

$$\min_{f \in \mathcal{F}} \mathcal{L}(f, D)$$

Most often, the family of functions will be characterized by a set of numerical **parameters**, Θ, which individuate its members:

$$\mathcal{F} = \{x \mapsto f(x;\theta) \mid \theta \in \Theta\}$$

Training is then finding the *values* of these parameters that minimize the loss value, while the overall structure of the prediction function remains the same. This is often done by finding the critical points of the loss function with respect to the parameters. For simple enough function families, a closed-form solution can be found. More often, however, iterative approaches have to be taken, beginning by some heuristically selected solution and repeatedly deducing a better one.

Some of these sets of options on which the training procedure operates may be closely related to each other, but their differentiating aspects may fall outside of the scope of this procedure. That is, they are "settings" of the predictive function on which the training procedure cannot operate. These aspects are known as **hyperparameters**. Naturally, different sets of hyperparameter values can be compared among each other on the same grounds as the functions that they each span. As such, automated procedures for their optimization exist. Some of these are even similar to the training procedure of their models. Thus, the terms "parameter" and "hyperparameter" can be seen as relative. Their principal role is differentiating what lies within the grasp of the training procedure.

In this thesis, we will refer to a particular function (a particular set of values for the parameters), however (sub)optimal, as a **predictor**. The family of functions will be referred to as a **model**. It should be noted that the term "model" is sometimes used in the literature to refer to a specific predictor rather than the set from which it is selected, as it is said to *model* the data. The term may also be used somewhat interchangeably. In the context of **neural networks**, models are often termed **architectures**.

1.1.2 Performance

1.1.2.1 Performance measures

In some contexts, regression especially, the end goal might be to minimize the loss directly or some quantity derived thereof. However, in many cases, the loss acts as a proxy for some other aspect of prediction wherein the practitioner wishes the predictor to perform well. These quantities of interest that characterize the prediction in some respect are known as **performance measures**. Performance measures may quantify some aspect of the discrepancies between predictions and expected outputs (in supervised learning) or they may simply provide some information about the predictions or the predictor itself.

As mentioned, oftentimes, in classification especially, the training procedure at work cannot directly operate on the performance measure of interest. Therefore, the practitioner must select an operable loss that will act as a proxy for it. For example, in classification, one might wish to achieve a low zero-one loss value. However, the zero-one function is not differentiable with respect to the prediction. Most training procedures will therefore not be able to operate on it directly.

If some specific performance measure is the primary quantity of interest to the practitioner, it can still be used to select a model or by a hyperparameter optimization procedure to select the best values for such hyperparameter. It is important, however, to remain mindful of the possible gaps between a satisfactory loss and a satisfactory performance as per the chosen performance measure.

Here are presented some common performance measures used in classification. The most simple of these is **accuracy**, the ratio of correct predictions to total predictions. Assuming some dataset, $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$, with $N \in \mathbb{N}$, and some predictor, f, let $\hat{y}^{(i)} = f(x^{(i)})$. Then the accuracy of f on D is given by:

$$\text{accuracy}(f, D) = \frac{\sum_{i=1}^{N} \mathbb{1}_{\hat{y}^{(i)}=y^{(i)}}}{N}$$

Accuracy is a proper classification performance measure in that it gives all classes equal standing. For this reason, its usage and values must be carefully considered, especially in settings where there is imbalance in the population size of each class. Indeed, as accuracy does not consider classes separately and simply counts the number of correct predictions, it can yield misleadingly high values to degenerate predictors that learn only the balance of classes without considering the input. In particular, a constant predictor outputting the majority class will obtain an accuracy equal to the dataset share of said class.

Other classification performance measures—often referred to as detection performance measures—distinguish certain classes, the ones to be *detected*. The most common ones, **precision**, **recall** and **f-measure**[1] concern themselves with a given *positive* class in a binary setting. Continuing the previous assumptions and further assuming $\hat{y}^{(i)}$, $y^{(i)} \in \{0, 1\}$, let 1 designate the positive class. Then, the precision of f on D is given by:

$$\text{precision}(f, D) = \frac{\sum_{i=1}^{N} \mathbb{1}_{\hat{y}^{(i)}=y^{(i)}=1}}{\sum_{i=1}^{N} \mathbb{1}_{\hat{y}^{(i)}=1}}$$

Likewise, the recall of f on D is given by:

$$\text{recall}(f, D) = \frac{\sum_{i=1}^{N} \mathbb{1}_{\hat{y}^{(i)}=y^{(i)}=1}}{\sum_{i=1}^{N} \mathbb{1}_{y^{(i)}=1}}$$

Both computations share a numerator, the number of observations correctly marked as positive (true positives). However, precision compares this count against the total number of positive predictions and recall, against the total number of positive instances. Therefore, the former is more concerned with how *discerning* a predictor is with respect to the positive class, and the latter, with how *sensitive* to it it is. These two concerns are somewhat at odds, as each would tilt a predictor to one side of a positive prediction. In other words, in doubt, maximizing precision would favor a negative prediction to avoid false positives, whereas maximizing recall would favour a positive one to avoid false negatives. To address this, f-measure seeks to capture the balance between them by measuring the harmonic mean between precision and recall. Given a balancing parameter $\beta \in \mathbb{R}_+$, the f-measure of f on D is given by:

$$\text{f-measure}(f, D; \beta) = (1 + \beta^2) \frac{\text{precision}(f, D) \cdot \text{recall}(f, D)}{\beta^2 \, \text{precision}(f, D) + \text{recall}(f, D)}$$

The value of β is the factor by which recall is favoured over precision. Its standard value is 1. Precision and recall, as well as f-measure, can be adapted to categorical settings, for example, by setting each class in turn as the positive class and aggregating the results. This stands in contrast with accuracy, which can be naturally used in categorical settings.

1.1.2.2 Computational complexity

Finally, another aspect that may be of interest in selecting models is their **computational complexity**. Indeed, like any function, a predictor will have time and space complexities associated with its computation. Furthermore, complexities associated with training and prediction might differ. It is important to consider all of these aspects, especially when computational resources are limited.

[1] Also termed f-score

For example, a practitioner might accept a slight decrease in performance performance measures in exchange for a great decrease in computational complexity, whether in training or prediction. This is usually a qualitative decision: no standard manner of balancing task performance against computational performance exists.

1.1.3 Generalization

Of course, the goal of exercising machine learning techniques is seldom to learn the set of observations at hand. The objective instead is to model the phenomenon underlying the data points collected. That is, we wish to be able to make a correct prediction for any imaginable point belonging to the same phenomenon. This is known as **generalization**: the predictor should be one that *generalizes* the patterns of the available data to the phenomenon at large, *i.e.* all the instances this phenomenon might produce. Of course, this is typically impossible to verify. The amount of possible instances may be infinite. Further, if some computable theory capturing the phenomenon of interest were available, there would be no need to apply machine learning towards it. Thus, there is usually no such theory available. Instead, common practice is to evaluate the trained predictor on a *sample* of observations, a **testing** set. Crucially, in order for this evaluation to be meaningful, the testing should not contain familiar examples, ones seen during training. In other words, the training and testing sets must be mutually exclusive. The performance of a predictor on the testing set then provides an estimate of how well the predictor performs in general.

Moreover, it is also common practice to have a **validation** set, separate from the training and testing sets. This validation set is also termed a development set. It serves to tune hyperparameters. Indeed, if the hyperparameters were to be adjusted in accordance with performance on the testing set, this would skew the predictor towards the particular sample that is the testing set, undermining its purpose. Hence, the testing set is to remain untouched until satisfactory performance has been attained on the validation set. Performance on the testing set is then a better indication of how well the predictor would perform on further observations that could be gathered in the future.

Validation sets are not to be confused with **cross validation**, another common evaluation technique. Cross validation consists in partitioning the set of available observations several times over (in parallel). Each of these partitions consists in a training and testing of its own. Each serves to train and evaluate its own predictor, resulting in as many predictors as there are partitions. Importantly, the previously described approach of having a single training set and a single testing set serves to indicate how well the particular *predictor* will generalize. In contrast, cross validation is used to gauge how well the *model* and approach at large will produce predictors that generalize well. Indeed, a particularly good (or bad) predictor could be the result of a particular training set. If one is interested only in deriving a good predictor, whether the underlying model is generally apt at producing them is no object. If, however, one is interested in the relevance of the approach at large, cross validation is a clever strategy to making the most of the same set of observation.

The first and most obvious obstacle to generalization is **underfitting**. Broadly, this describes a situation in which a predictor cannot adapt to the data at hand and produces poor results even for the data that it is trained on. Without fitting the available data, the predictor has little hope of fitting the underlying phenomenon. The customary example of such a scenario is attempting to fit a linear model to a non-linear function. Of course, the resulting predictor may suffice as a crude approximation in some situations. Nevertheless, underfitting is generally undesirable.

A more subtle and widely discussed occurrence is **overfitting**. As the term suggests, this happens when a predictor overtly adapts to the particulars of the training set. These particulars may come from noise in the collection process or external phenomena separate from the phenomenon to be modelled. Thus, fitting these disturbances as part of the predictor can hinder its ability to properly generalize said phenomenon of interest. Informally, the predictor is said to "learn by heart" because it captures what it has been shown as opposed the underlying phenomenon. An example of overfitting would be the converse of the previous example: fitting a non-linear model to a linear function. Noise in the data might cause the predictor to incorporate these disturbances into a non-linear predictor. Figure 1.1 illustrates overfitting and underfitting.

Underlying these notions—underfitting and overfitting—is a crucial one: **capacity**, the wealth of functions covered by a model. For example, a polynomial of a given degree can be fit to a lower degree one by setting the appropriate coefficients to zero. The converse is not true. Thus, a higher-degree polynomial can *represent* more functions than a lower-degree one. Capacity is often discussed informally and intuitively. Nonetheless, formal definitions exist, such as the Vapnik–Chervonenkis dimension (see Goodfellow et al., 2016).

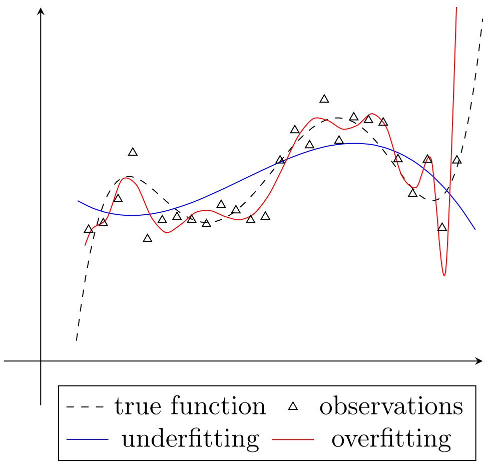

Figure 1.1 An example of overfitting and underfitting. The observations collected from the underlying true function mislead the overfitting function. The underfitting function does not capture the data nor the true function.

Of course, the ideal model capacity for a given setting is seldom known in advance. Practitioners may then test models of varying capacity. Alternatively, if the practitioner intuits the capacity of a model to be too large, they may employ regularization, which is presented in the following section.

1.1.4 Regularization

A common approach to mitigating overfitting is **regularization**. Broadly construed, any amendment to the training procedure to the end of reducing overfitting will fall under the umbrella of regularization. More precisely, regularization will often seek to favour certain regions of the solution space *a priori*. That is, while the training procedure is designed to favour regions of the solution space that fit the data, regularization will favour or disregard certain regions regardless of the data. This may appear counterproductive, but reducing the solution space may further define a problem that is underdefined. Although regularization does not reduce the actual capacity of the model at hand, its realizable capacity is hindered in the sense that some solutions become less attainable. Perhaps a more intuitive interpretation is that regularization acts as a penalty—or smooth constraint, artificially increasing the difficulty of the task. In turn, this reins in the ability of the training procedure to produce predictors that are overtly suited to the training data.

Regularization comes in many forms. Some are quite general, while others are more specific to certain models or training procedures. Notable examples are parameter norm penalization and early stopping. Other regularization methods will be discussed in the context of neural networks (Section 1.2.5). Parameter norm penalization simply counts the norm of the parameter vector as part of the loss incurred by a set of parameter values. Given a norm function, Ω and a hyperparameter, $\lambda \in \mathbb{R}_+$, weighting the importance of the norm, the amended loss function becomes:

$$\tilde{\mathcal{L}}(f_{\boldsymbol{\theta}}, D) = \mathcal{L}(f_{\boldsymbol{\theta}}, D) + \lambda\Omega(\boldsymbol{\theta})$$

This explicitly encourages the training procedure to avoid larger parameter values, with λ throttling the degree of avoidance. Naturally, this behaviour is also influenced by the choice of the norm, Ω. By and large, the most common norms used as regularizers are L^p norms[2], in particular L^2 and L^1. The L^2 norm will act more smoothly, punishing (absolutely) large parameter values. In contrast, the L^1 norm will encourage sparser solutions, *i.e.*, solutions where more parameters are set to zero (Goodfellow et al., 2016).

Another ubiquitous form of regularization is **early stopping**. Unlike parameter norm penalties, it exclusively applies to iterative training procedures. Iterative procedures will attempt to find a solution by repeatedly moving to a neighbouring solution with a reduced loss value. In so doing, these ameliorations may move into a regime where the training loss declines but generalization deteriorates. As its name implies, early stopping will halt the training procedure when this begins to happen, as gauged by a validation set. This probing of validation performance can be done as per the loss or some other performance performance measures. The appeal of early stopping is obvious: not only is it simple to implement but it can reduce computation time in a performance-forward manner by foregoing training that may be superfluous.

1.1.5 Prior knowledge

Throughout Section 1.1, the crucial role of prior knowledge has become increasingly apparent. Indeed, incorporating prior beliefs about the nature of the phenomenon that the model is meant to capture into the model can greatly improve results. This seems to imply that there exists no one-size-fits-all approach to machine learning. This notion is formalized by the *No Free Lunch* theorem (Wolpert, 1996), which states that, when averaged over all possible data-generating distributions, all algorithms perform equally well on unseen observations. In other words, no algorithm is better than any other overall. This implies that the *assumptions*—the prior knowledge

[2] Given some vector $\boldsymbol{x} \in \mathbb{R}^m$, its L^p norm, $p \geq 1$, is given by: $\|\boldsymbol{x}\|_p = (\sum_{i=1}^m x_i^p)^{\frac{1}{p}}$

administered—are the key to finding a satisfactory predictor, rather than attempting to find a broadly more powerful model. Prior knowledge can take many forms.

One form that prior knowledge can take is the base representation of the data. Indeed, when the practitioner knows which aspects of the data are relevant and how to express them, the problem the predictors must solve is greatly simplified. Moreover, the more refined an informed representation is, arguably, the more succinct it will become. *A contrario*, when the relationship between the components of observations is not known, they will be left separate. Hence, the cruder the base representation of the data at hand, the higher its dimension. This creates a significant problem because the number of configurations grows exponentially with respect to the dimension of the input space, making the data increasingly sparse. This is known as the **curse of dimension**. It makes learning difficult because the number of regions of the input space increases much more quickly than the number of observations usually can, causing these regions to be empty.

Nonetheless, if no prior-informed base representation is available—as is often the case in machine learning—one can infer such a representation by posing broad and more abstract assumptions about how the data behave. This is the case of representation learning through neural networks. While it may seem overtly ambitious to pose assumptions with great reach, especially in light of the *No Free Lunch* theorem, the two core tenets underlying neural networks are reasonable enough that it is quite easy to ascribe them to many problems. The first is the manifold hypothesis. Broadly, this states that most of the data space is typically empty, with observations being concentrated in a small connected space, the manifold. Further, this space is locally Euclidean, so that within this space the variations between observations are fairly smooth. The second tenet is that the coordinate system of this manifold can be built by repeatedly composing factors, beginning in the original space. That is, starting with the base representation, one can build increasingly complex intermediate representations until arriving at the representation where the data is amenably expressed, where small shifts map to valid observations. These simple assumptions can be quite powerful. Through them, neural networks can build useful representations.

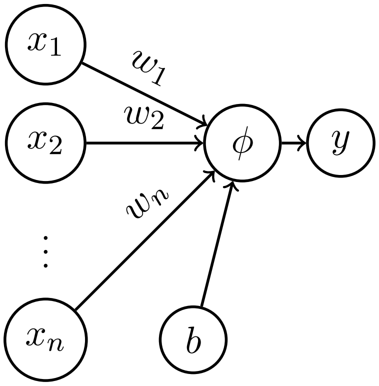

Figure 1.2 An artificial neuron. The weight parameters adjust the importance of each component of the input, $\boldsymbol{x}$, in their sum. The bias term, b, offsets this sum. An activation function ϕ is applied on the result.

1.2 Neural networks

Neural networks are built from artificial neurons. Inspired by models of cognition of the time, these artificial neurons were conceived as a straightforward prediction function. As illustrated in Figure 1.2, given an input vector, $\boldsymbol{x}$, one can adjust the importance of each of its components via **weights**, $\boldsymbol{w}$, which multiply them. A bias term, b, offsets this sum. The result is fed through an **activation function**, ϕ:

$$\boldsymbol{x} \mapsto \phi(\boldsymbol{w}^\top \boldsymbol{x} + b)$$

Historically, the basis for neural networks are the perceptron and ADALINE models (Rosenblatt, 1957; Widrow, 1960). These correspond to setting the activation function as, respectively, a threshold function and the identity (often termed linear activation).

As linear models, these artificial neurons are fairly limited. One can remediate this impediment by adding intermediate—also called hidden—**layers** of **units**. As shown in Figure 1.3, each unit (neuron) of this hidden layer receives an activation from each input unit individually weighted. These activations are summed together with a bias term and fed to the layer activation function. The output is then sent to each unit of the following layer. This is termed a Multi-Layer Perceptron or **Feed-Forward Network (FFN)** (Goodfellow et al., 2016).

Many hidden layers can be added in this manner. Their computation can be conveniently effected as an affine transformation. Given the value of the i-th layer, $\boldsymbol{a}^{(i)}$ the $(i + 1)$-th layer, $\boldsymbol{a}^{(i+1)}$ is

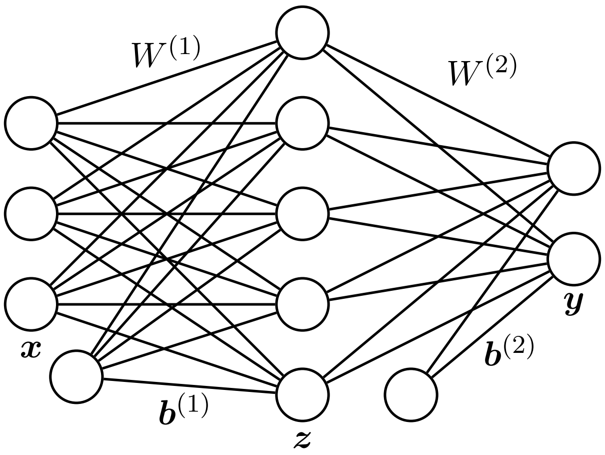

Figure 1.3 A feed-forward network with a single hidden layer mapping $\boldsymbol{x}$ to $\boldsymbol{y}$. Its parameters are the weight matrices $W^{(1)}$ and $W^{(2)}$ and the bias vectors $\boldsymbol{b}^{(1)}$ and $\boldsymbol{b}^{(2)}$.

computed as:

$$\boldsymbol{a}^{(i+1)} = \phi(W\boldsymbol{a}^{(i)} + \boldsymbol{b}),$$

where weight matrix, W, and bias vector, $\boldsymbol{b}$, are learned parameters, and ϕ denotes the selected activation function.

Remarkably, FFNs are universal approximators (Hornik, 1991). That is, any computable function can be approximated to an arbitrary degree of precision by some FFN with a single hidden layer of appropriate size. There are some caveats to this result. Firstly, not any FFN will be able to approximate any function. Secondly, the universal approximation property does not imply that the function can be *learned*. Nonetheless, it shows the representation power of neural networks.

1.2.1 Back-propagation

As previously mentioned, when handling complex function families, closed-form solutions are rarely possible. Instead, iterative approaches to optimization need to be taken. **Gradient Descent (GD)** (Cauchy, 1847) is a well established option. Beginning with some heuristically selected set of values for the function parameters, GD will compute the gradient of the loss function with respect to these parameters. Then, these parameters will be modified slightly in the direction opposite to the gradient as per a hyperparameter termed the **learning rate**. In doing so, the loss will be ameliorated at the local scale. From these improved values, the process can be repeated, eventually arriving at satisfactory parameter values. Naturally, GD is not guaranteed to arrive at a globally optimal solution in the general case: improvement can halt at flat regions of the solution space. Nonetheless, this is not often considered an issue for two reasons. On one hand, optimal performance on the training set can be synonymous with overfitting: there is no sense in seeking the best set of parameter values as per the training data if it will generalize poorly. On the other hand, an optimal loss will not necessarily entail an optimal value in the performance performance measure of interest.

Neural networks are trained by **Stochastic Gradient Descent (SGD)** (Bottou et al., 1998), a variant of GD wherein the gradients on the entire training set are approximated by computing the gradients on a sample of observations, a **minibatch**—henceforth referred to as a **batch**[3]. This requires the computation of the gradient with respect to each weight in the network for each observation in each batch. This can amount to a heavy computational burden for large networks with large parameter counts. More importantly, if performed naively, this burden can be exacerbated by significant redundancies. Indeed, when computing the gradient of the loss with respect to a given weight, one must—perhaps implicitly—compute the gradients with respect to all the weights that follow this weight along the network. In the example in Figure 1.3, computing the gradient of the weight matrix $W^{(1)}$ with respect to the loss requires computing the gradient of $W^{(2)}$. It is therefore more efficient to compute the gradients in *reverse*: beginning with the last layer and moving backwards along the network, using the previously computed gradient to compute the present one. This is the intuition behind the **back-propagation** algorithm (Rumelhart et al., 1986): leveraging the chain rule of derivation to expedite the computation of the gradient of the entire neural network. It also has a quite natural interpretation: After propagating, the activation forwards through the network. The discrepancies between the actual and expected outputs provide the network with corrections to effect. Each neuron can then pass along its correction to each of its antecessors.

1.2.2 Activation functions

In spite of their advantages, deep neural networks can be difficult to train. Some of these difficulties, such as overfitting, are to be expected in that they are habitual to high-capacity models These can be addressed by regularization, which will be expounded in Section 1.2.5. Other issues are specific

[3] Some authors favour the term "minibatch" to avoid conflation with standard GD, which is also termed Batch Gradient Descent. Nevertheless, the term "batch" is more prevalent and will be the one used in this thesis.

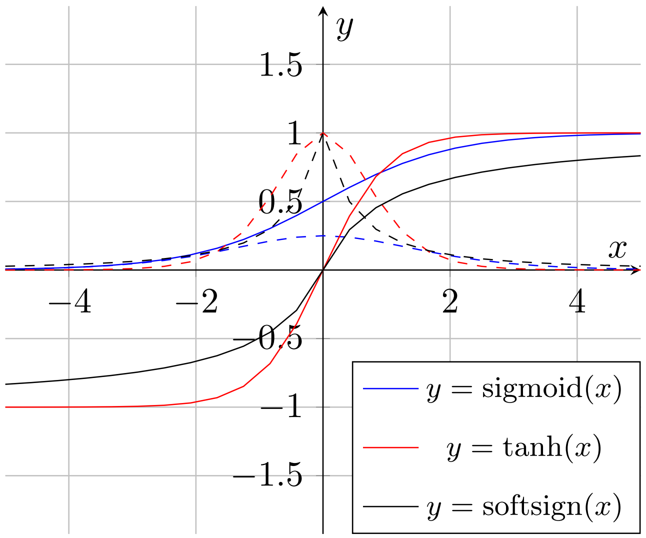

Figure 1.4 Common "squashing" activation functions used in neural networks and their derivatives (in dashed lines).

to the nature of deep neural networks, in particular, their depth. The use of so-called squashing activation functions—functions that saturate at both ends of their domain—in the hidden layers of FFNs confers them with powerful non-linear properties. Nonetheless, these can cause some training issues. As shown in Figure 1.4, the derivatives are null over most of their domain. To boot, the derivative of the sigmoid function is at most $\frac{1}{4}$. Because the gradient of early layers depends multiplicatively on the values of these derivatives, it can become so small in norm as to impede learning altogether. More intuitively, as the corrections to effect propagate back through the network, they become smaller—even disappearing through numerical limitations. All in all, this is consistent with the premise of these activation functions: their regime of influence lies in a small interval. Beyond these bounds, they are flat, their input alters their output little. Therefore, instructing the portions of their network that precede them to adjust would have little bearing on the overall output of the network, for better or worse. Dissipating gradients can be largely mitigated by opting for different activation functions: ones that saturate only at one end of their domain. Examples of such activation functions are **Rectified Linear Unit (ReLU)** (Glorot et al., 2011), **Exponential Linear Unit (ELU)** (Clevert et al., 2016) and **softplus** (Dugas et al., 2001), which are illustrated in Figure 1.5. These functions are widely used in the hidden layers of very deep networks. Indeed, as shown in Figure 1.5, these functions are increasing throughout the positive reals, making their derivatives non-zero. This allows for better gradient flow in deep networks.

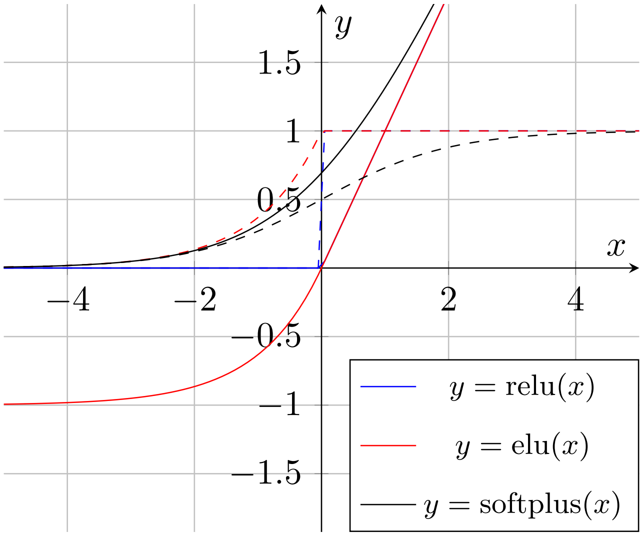

Figure 1.5 Common "non-squashing" activation functions used in neural networks and their derivatives (in dashed lines)

1.2.3 Parameter initialization

Another concern in training deep neural networks is the initialization of their parameters. Indeed, as an iterative algorithm, SGD will repeatedly attempt to improve on its present solution, beginning with some initial solution. As such, this first set of values for the predictor parameters can have significant influence on which solutions can be attained. Unfortunately, there is little understanding of how any specific set of values will affect the performance of the solutions that might emerge from it, both in terms of the training data and generalization.

Consequently, initialization heuristics for neural networks focus on conferring the layers with properties that are thought to aid initial learning (Goodfellow et al., 2016). One important aspect of this is differentiating the units in a layer from each other in order to avoid redundancy (breaking symmetry). To mitigate this, weights are initialized randomly, usually following a normal or uniform distribution. An additional consideration is that, as mentioned in Section 1.2.2, the range of activation of non-linear units is bounded (at least on one end). If weight initialization puts the pre-activation totals in the regions of saturation, changes to those weight will carry little information, slowing down initial learning. A first step towards avoiding this is centring random initialization around zero.

Still, the weights inbound to a unit can bring the unit into saturation by sheer force of their range of values, even when these are diverse and centred around zero. Yet, scaling down weight initialization to some arbitrarily small range could mitigate the symmetry-breaking effects. Instead, they can be scaled down according to their *number*. That is, as the number of inbound connections increases, the variance of their weights is set to decrease. In essence, this corresponds to maximizing variance as much as possible while shying away from saturation.

A similar but converse situation can be observed in back-propagation and addressed in a similar manner. However, a given weight will be tied to a different set of weights in back-propagation. Indeed, if a weight is to contend with the weights of other connections with which it shares its arrival, it must also contend with the weights of connections with which it shares its departure. The most prevalent initialization strategies therefore compromise between these two requirements (Glorot & Bengio, 2010; He et al., 2015). As for biases, given that they are exclusive and unique to a unit, they need not be initialized randomly but rather to a heuristically selected constant.

1.2.4 Stochastic Gradient Descent variants

An important issue with SGD is that convergence towards a good solution can be slow when there is high variance in the gradients corresponding to each batch. That is, if the gradients of consecutive batches contradict each other in several components, much of the updates parameters will be wasted going back and forth along the same lines. To mitigate this, one can amend the SGD algorithm to account for the overall trend of updates by keeping a running average of gradients. This is termed **momentum**. For conciseness, for some batch, $\mathcal{B} = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(b)}, y^{(b)})\}$, loss function, ℓ, and predictor, f, parameterized by $\boldsymbol{\theta}$, let:

$$\tilde{\mathcal{L}}(\boldsymbol{\theta}, \mathcal{B}) = \frac{1}{|\mathcal{B}|} \sum_{i=1}^{b} \ell(y^{(i)}, f(x^{(i)}; \boldsymbol{\theta})).$$

In standard SGD the parameters of the predictor, $\boldsymbol{\theta}$, are updated directly from the gradient:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \epsilon \nabla_{\boldsymbol{\theta}} \tilde{\mathcal{L}}(\boldsymbol{\theta}, \mathcal{B}),$$

where ϵ denotes the learning rate. With momentum, the gradient serves instead to update the running average of gradients, $\boldsymbol{v}$:

$$\boldsymbol{v} \leftarrow \alpha \boldsymbol{v} - \epsilon \nabla_{\boldsymbol{\theta}} \tilde{\mathcal{L}}(\boldsymbol{\theta}, \mathcal{B}),$$

where α is the momentum hyperparameter. The update to the parameters is then applied from this average:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \boldsymbol{v}.$$

Alternatively, one can compute the gradients given the present value of this running average as a "look-ahead":

$$\boldsymbol{v} \leftarrow \alpha\boldsymbol{v} - \epsilon\nabla_{\boldsymbol{\theta}}\tilde{\mathcal{L}}(\boldsymbol{\theta} + \boldsymbol{\alpha v}, \mathcal{B}).$$

The parameter update remains the same.

Another issue with training neural networks with SGD is that the learning rate is shared across parameters. Finding a learning rate that is adequate for all parameters is difficult. Similarly, manually tuning a learning rate for each individual parameter is impracticable even for models of modest size (1k parameters). Consequently, a few SGD variants have been proposed that adapt the learning rate for each parameter automatically. This adaptation is based on the gradient history. The most successful of these algorithms, Adam ("adaptive moments") (Kingma & Ba, 2014), uses the exponentially-weighted first and second moments of the gradient history.

1.2.5 Regularization

In practice, neural networks tend to have high capacity. Large amounts of data are not always readily available, making regularization highly solicited. Both parameter norm penalization and early stopping, discussed in Section 1.1.4, are commonly applied to neural networks. There exist regularization strategies that are more specific to neural networks and the back-propagation framework. This section will go over two important ones, namely, parameter sharing and dropout.

Rather than explicitly constraining parameter values through the use of a norm penalty, one can constrain parameters by using them for several different problems. This is the essence of parameter sharing. Not only is the solution space reduced by implicating the same parameters in several aspects of the loss, it can also administer prior knowledge in so doing. Parameter sharing can take many forms. One is to repeat the same parameters in different parts of a model (Bengio et al., 2013; Ravanbakhsh et al., 2017). This constrains the training procedure to find a value for a shared parameter that adequately fulfills its multiple roles. In GD, the gradient of the loss with respect to this parameter will be informed by each operation in which it partook. Moreover, this

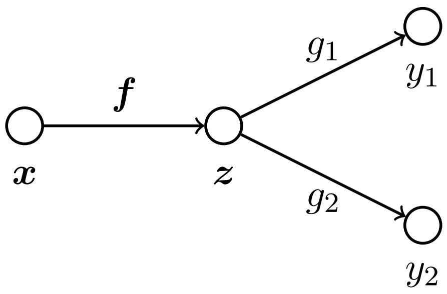

Figure 1.6 In multitask learning, models will share some parts. In this example, the models $g_1 \circ \boldsymbol{f}$ and $g_2 \circ \boldsymbol{f}$ share $\boldsymbol{f}$ and the resulting representation of $\boldsymbol{x}$, $\boldsymbol{z}$. The training procedure must therefore produce a choice of $\boldsymbol{f}$ that satisfies both tasks.

explicitly decreases the capacity of the model when compared to an equivalent one with no repeated parameters. Another form of parameter sharing is to have a model or parts thereof participate in several tasks (Bengio et al., 2013; Erhan et al., 2010). This is termed **multitask learning**. An example of such a scenario would be a set of data with observations having several targets involving separate loss functions but sharing a common input. The model for each task would require its own output, they can share a common neural network learning a representation that is adequate for both tasks. This is illustrated in Figure 1.6. Training can be performed jointly or in succession. Semi-supervised learning is a special case of multitask learning. It is a frequent occurrence that much more unannotated than annotated data is available. The unannotated data can be used in some unsupervised task with a neural network. This neural network can then be re-purposed for the supervised task on the annotated data. This is often termed **pre-training**.

Another common regularization method in neural networks is **dropout** (Hinton et al., 2012; Srivastava et al., 2014). To apply dropout with GD, at each iteration step, non-output units are randomly set to zero—"dropped"—when computing the loss. The sampling of this dropping is usually conducted independently for each unit. Moreover, the probability of dropping units—a hyperparameter—can be set independently for each layer. Dropout is meant to prevent co-adaptation between units. That is, by intermittently suppressing units upstream and downstream, it encourages units to create features that are useful overall rather than relying on other units to correct their mistakes (Srivastava et al., 2014). Dropout can also be interpreted as approximating the averaging together of the predictors that would result from all the combinations of unit removals (Hinton et al., 2012).

1.2.6 Beyond feed-forward networks

Some data are formatted in such a way that they benefit from specialized neural network structures over standard FFNs. In these neural networks, the base building block remains the parametric affine transformations and non-linear activation functions introduced at the beginning of this section. These are known in practice as **dense layers**. Other operations might be added along the network for specific types of input. Some of these operations can also be combined by higher-order functions. Indeed, many diverse combinations are possible, and much work in the area is devoted to the design and study of which such structures are beneficial to different types of data. One crucial requirement in building these networks is that all functions constituting them must be differentiable with respect to their inputs throughout their domain. This allows for backpropagation through the entire network and the joint learning of its parameters from a singular loss. This notion gives rise to the term "**differentiable programming**". Indeed, one can piece together a series of arbitrary, differentiable operations—even parameter-free ones—and jointly learn them via back-propagation. This section will go over two of the most important specialized neural networks: convolutional and recurrent neural networks.

In order to map large, structured observations such as an image to a single output, one would require to aggregate the transformations applied to each pixel. A global aggregation would focus on immediate, shallow aspects, drowning out local patterns. Instead, **Convolutional Neural Networks (CNNs)** (LeCun, 1989) seek out these local patterns by applying transformations on local segments of the input. Crucially, these transformations are performed with the same parameters over all segments. The segments are often set to overlap, as illustrated in Figure 1.7. This parameter sharing makes CNNs very powerful by naturally accounting for shifts in the position of patterns. As shown in Figure 1.7, layers of convolution with different parameters can be applied in succession with later layers relying on patterns detected by previous layers to build more complex ones (Goodfellow et al., 2016).

Another case requiring specialized neural networks are inputs of indefinite size. Trimming and padding with dummy values is a possible approach. Nonetheless, these types of data often have or can be assigned a recursive graph structure. This can be used to leverage parameter sharing and naturally account for variable size. This is what **recursive neural networks** do: They traverse the graph in topological order applying the same function at each node. This function takes both

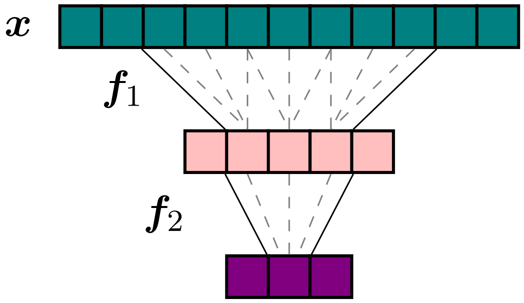

Figure 1.7 CNNs apply a transformation on (potentially) overlapping segments, allowing the shift-invariant learning of local patterns. Further, the composition of convolutions allows for more complex patterns as well as a larger capturing field for downstream elements. The elements of $\boldsymbol{f}_2 \circ \boldsymbol{f}_1(x)$ depend on a larger section of $\boldsymbol{x}$ through $\boldsymbol{f}_1(x)$.

the values of the nodes and its previous outputs as arguments.

The most common form of recursive neural network are **Recurrent Neural Networks (RNNs)**, which are used on chains, *i.e.* sequential data (Jordan, 1986). As illustrated in Figure 1.8, RNNs read the elements of the input sequence in order, updating their **hidden state** as a function of them. The function updating this hidden state is termed the **transition function**. The initial hidden state can be learned or set to a constant, usually null. Depending on the use-case, one may be interested in all the hidden states or just the last one, which is a function of the entire sequence. For example, in predicting the weather for the next day based on that of the current and previous days, the hidden state at each time-step would serve as its own output. In contrast, in predicting the overall sentiment of a movie review from the sequence of words constituting it, only the last hidden state would contain all the relevant information. Like with CNNs, multiple RNNs can be applied in succession, each one taking the sequence of hidden states of the previous one as its input. This allows the networks placed higher in the hierarchy to operate on larger timescales (Hermans & Schrauwen, 2013). Additionally, two RNNs can be applied separately in opposite orders on the sequence, allowing to analyse the information flow in both directions. This is known as a Bidirectional RNN (Schuster & Paliwal, 1997). RNNs are a mainstay of natural language processing, the subject of the next and final section of this chapter.

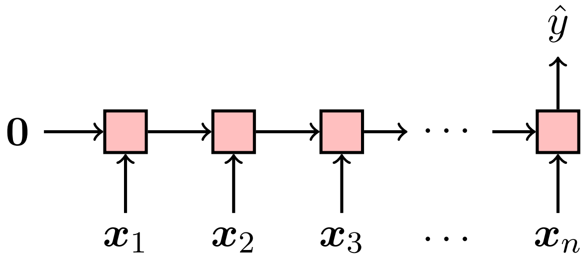

Figure 1.8 An RNN processes the tokens sequentially, updating its hidden state, represented by the pink squares.

1.3 Natural Language Processing

The field of AI concerned with human language is termed **Natural Language Processing (NLP)**. Needless to say, language is extremely complex. One of the key issues in applying machine learning to language is representation. Language utterances must be presented as mathematical objects on which machine learning algorithms can operate. Sequences—or even sets—of symbols, are a simple and natural choice. However, from that first step a few issues arise. Firstly, one must define what will constitute a symbol. This segmentation of the string of characters is termed the **tokenization**. A reasonable option would be words or characters. This can nevertheless lead to some difficulties. Most majorly spoken languages count their vocabularies in the hundreds of thousands, incurring a significant computational burden: if a predictor must account for each word separately, it will require large parametrization to store what it has inferred about the occurrences and co-occurrences of these words. Further, the rate of occurrence between words is notoriously uneven, in any language. Given a sufficiently large corpus, in compiling a list of words in decreasing order of number of occurrences, one can observe that the k-th word would appear roughly $1/k$ as many times as the most frequent word. This is known as Zipf's law (Manning & Schutze, 1999). It has been observed across a variety of human languages, including artificial ones, as illustrated by Figure 1.9. As it relates to machine learning, this skew further worsens the curse of dimension because words added from the tail end of the vocabulary will be much rarer than common ones.

Common practice involves restricting the vocabulary based on the **corpus** involved in the task. This can be done by setting a fixed size for the vocabulary or disregarding words outside a set range of total occurrences or proportion of documents covered. Nonetheless, this can result in the exclusion of occurrences of what might arguably be vocabulary words through conjugation, alternate spellings,

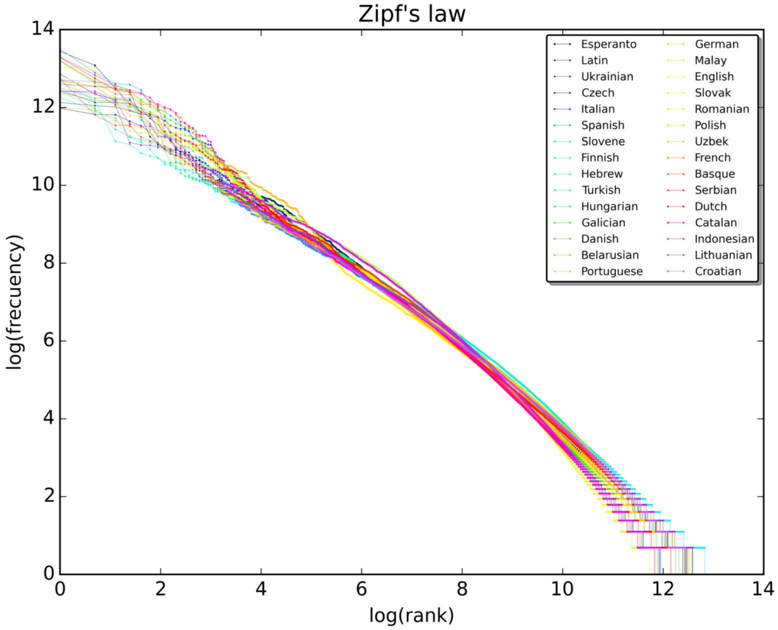

Figure 1.9 Illustration of Zipf's law across multiple languages (from: `https://commons.wikimedia.org/wiki/File:Zipf_30wiki_en_labels.png`)

misspellings or obfuscations (*e.g.* `w0rd` instead of `word`).

Secondly, modelling utterances as sequences of symbols posits the operable space to be the set of *all* possible sequences of symbols. Even when restricting these to some finite length, the space can be quite large. More importantly, the effective space of utterances that might occur is extremely small in comparison. Not only can a great many words appear rarely, as previously mentioned, but most combinations would not constitute viable phrases grammatically, still less semantically. Of course, this issue is only aggravated when proceeding with characters rather than words as the base unit on which sequences are built.

These issues are addressed by two broad strategies: simplifying the base representation, and mapping the base representation to a learned denser one. These approaches can also be combined. While simplification will obviously apply an incorrect bias to the resulting predictors, it can greatly improve the computational and data efficiency of the approach. Dense representations allow downstream tasks to operate in a smaller space. Nonetheless, the learning of these representations will often require exercising unsupervised learning. In turn, these tasks will also impose their own biases, as they implicitly establish what aspects of utterances are important.

The remainder of this section will introduce basic notions and challenges of NLP, namely approaches to structuring utterances from words and characters as well as strategies to representing these. To avoid ambiguity and to address different levels of tokenization (*e.g.* word, character), the possible symbols will be referred to as **terms**, and the occurrences of such symbols will be referred to as tokens.

1.3.1 Bags of words

One manner of simplifying the base representation of utterances is to treat them as sets—rather than sequences—of tokens. This was largely the dominating approach until recent years.This approach alleviates the combinatorial explosion of the base domain by eschewing token order. It also makes vectorization trivial. Indeed, given the **vocabulary**, V, utterances become vectors in $\{0,1\}^{|V|}$, with each component indicating the presence or absence of a term. This is termed a **bag-of-words** representation because it does not structure the tokens in an utterance in any way. Commonly, bag-of-words representations will be based on multisets rather than sets, allowing to account for the number of occurrences of terms, as opposed to just their presence. Vectorization remains simple, with utterances being vectors in $\mathbb{N}^{|V|}$. In either case, in order to promote numerical stability, the vectors can be normalized in some manner. The most common form of normalization is L^1 normalization. In such a way, one can conveniently and efficiently encode large amounts of text in a consistent and stable manner.

However, as the amount of text comprised in single vectors grows, frequent tokens can drown out rarer ones. To boot, when the order of tokens is ignored, rarer terms are arguably the hinge pin of discernment between documents. Therefore, common terms are often sought to be curtailed. A class of words that are often discarded are **stop-words**, also termed function words (Manning & Schutze, 1999). These include prepositions and determinants. These words play an important grammatical role but do so with respect to neighbouring words. When abandoning word order, any notion of neighbourhood is lost. Hence, they are often deemed unhelpful.

A more data-dependent strategy is to remove frequent terms, *e.g.* based on rank, or document frequency—the number of documents in the dataset containing one or more occurrences. A smoother approach is to weight terms by some function that is decreasing in their frequency. One notorious technique is to weight terms decreasingly with their document frequency. This is known as **Inverse Document Frequency (IDF)** (Spärck Jones, 1972). Given the number of documents considered, N, the IDF weight assigned to term t is given by:

$$\mathrm{IDF}(t) = \log \frac{N}{n_t},$$

where n_t is the number of documents where t is present.

Nevertheless, the most conspicuous weakness of bag-of-words representations is not accounting for token order. This is a particularly pronounced issue for **analytic** languages, where meaning is established through word order rather than word declinations. Further, for character-level tokenization, bag-of-words are all but unviable, as words become completely scrambled.

To alleviate this issue, common practice is to extract sequences of tokens of set lengths from the data. These are collectively known as ***n*-grams**[4], with the use of "n" intimating the variability of their length. To mitigate instability due to incidental shifts, *all* sequences of the chosen length will be extracted, as opposed to segmenting each document once. For example, the following sentence would be tokenized into word bigrams thusly:

$$\texttt{the cat sat on the mat} \mapsto \{\langle\texttt{the},\texttt{cat}\rangle, \langle\texttt{cat},\texttt{sat}\rangle, \langle\texttt{sat},\texttt{on}\rangle, \langle\texttt{on},\texttt{the}\rangle, \langle\texttt{the},\texttt{mat}\rangle\}.$$

Further, a *range* of lengths can be extracted instead of a single length. Although, in practice, the number of possible n-grams does not grow as quickly with respect to n as it could (exponentially), it still can grow quite rapidly. Hence, for word n-grams, common (non-unit) lengths used in practice are two (2) and three (3). In contrast, longer character n-grams can be extracted (Manning & Schutze, 1999). In any case, n-grams become terms unto themselves. They can thereafter undergo the same treatment as singletons, such as being filtered out or weighted in accordance with their frequencies.

Off-the-shelf machine learning models can operate directly on bag-of-words representations for a variety of tasks. Alternatively, bag-of-words can be transformed into more sophisticated ones by approaches imposing certain priors. An example of this are **Latent Dirichlet Allocation**

[4] For lengths one (1), two (2) and three (3) the terms commonly employed are, respectively, "unigram", "bigram" and "trigram."

(LDA) (Blei et al., 2003) and its numerous variants (Perotte et al., 2011; Teh et al., 2006). These algorithms are used to perform **topic modelling**, the grouping of words into topics and the detection of those groups in documents. LDA posits topics as categorical distributions over the vocabulary of words given a Dirichlet prior.[5] As such, it operates on the unnormalized multiset bag-of-words representation. Once trained, it assigns each document a distribution of probability over the inferred topics. Thus, the sparser bag-of-words representation can be transformed into a denser, more informative one by smoothing its tokens into cognate ones—as per their co-occurrences.

1.3.2 Sequences of words

In natural language, syntax often requires words to be ordered in a specific manner for the resulting phrase to be grammatically sound. Moreover, two different orderings of the same set of tokens can convey contrasting meanings. Thus, it can be desirable to treat utterances as sequences—rather than sets—of tokens. This is especially true when the tokens considered are characters or short sequences thereof, like syllables.

To this end, the position of each token must be encoded into its representation. This can be done through several different approaches. Perhaps the most elegant one is to construct it *implicitly*. That is, to let the order of processing of the sequence reflect the order of tokens. This is the approach taken by RNNs (see Figure 1.8). Due to the non-linear nature of the transition function, consuming a sequence of tokens in different orders will result in a different hidden state.

However, some issues can arise from parsing an utterance sequentially. It is common for two tokens relating to each other to be separated by several other tokens. This is often termed a **long-range dependency**. In terms of back-propagation, this means that information from the gradient from a later token must traverse the RNN backwards in order to inform the gradient from an earlier token. As mentioned in Section 1.2.2, gradients can vanish when traversing deep networks. This is particularly likely in RNNs, as the parameters involved in the transition function are always the same from token to token. The transition function of a simple RNN computes the hidden state at

[5] The Dirichlet distribution is the multivariate extension of the Beta distribution. Its realizations can be interpreted as categorical distributions.

time step, t, $\boldsymbol{h}^{(t)}$ given the input at that time step, $\boldsymbol{x}_t$, as:

$$\boldsymbol{h}^{(t)} = \tanh(W\boldsymbol{h}^{(t-1)} + U\boldsymbol{x}_t + \boldsymbol{b}),$$

where W and U are weight matrices, and $\boldsymbol{b}$ is a bias vector. The gradient of the hidden state over n steps is given by:

$$\frac{\partial \boldsymbol{h}^{(t)}}{\partial \boldsymbol{h}^{(t-n+1)}} = \prod_{i=t-n}^{t} \operatorname{diag}(\tanh_i')W,$$

where $\tanh_i'$ is the derivative of the activation at step i. This heavy dependence on W can cause gradients to vanish—or indeed to explode—unless W has certain properties (see Pascanu et al., 2013), making long-range dependencies difficult to learn. This can be mitigated by variant RNNs using gate mechanisms that allow more stable gradient flow (see Cho et al., 2014).

Another approach to preserving the order of tokens is to explicitly compute some encoding of its position and integrate it to the term representation. A straight-forward approach is to have a randomly initialized parameter vector per possible position (Gehring et al., 2017). This has the obvious limitation of posing a hard limit on the number of positions that can be handled by the predictor. Further, these vectors are handled symbolically: they are simply defined to be different from each other, with no notion of distance between them. As such, there is no mechanism in place to transpose what has been inferred at some position to others. Thus, an unseen shift in an utterance would be completely unfamiliar to the predictor.

A more expressive encoding is sinusoidal position encodings (Vaswani et al., 2017). The vector corresponding to a position is computed by a sine function, with each component either having a different period or offset. Let $d, M \in \mathbb{N}$ be the dimension of the position vector and a normalizing hyperparameter, respectively. The m-th component of position i is given by:

$$\boldsymbol{p}(i)_m = \begin{cases} \sin\left(\frac{i}{M^{m/d}}\right) & 0 \equiv m \mod 2 \\ \cos\left(\frac{i}{M^{(m-1)/d}}\right) & 1 \equiv m \mod 2 \end{cases},$$

This is illustrated in Figure 1.10. While some component values will repeat, by choosing a sufficiently large M, there will be no collisions between positions in practice. Furthermore, the difference between positions can be expressed as a linear map.

Thus far, the encodings have been applied to absolute positions. This is sensible in encoders aggregating a sequence of tokens to a single vector. However, in non-aggregating encoders, *i.e.* encoders

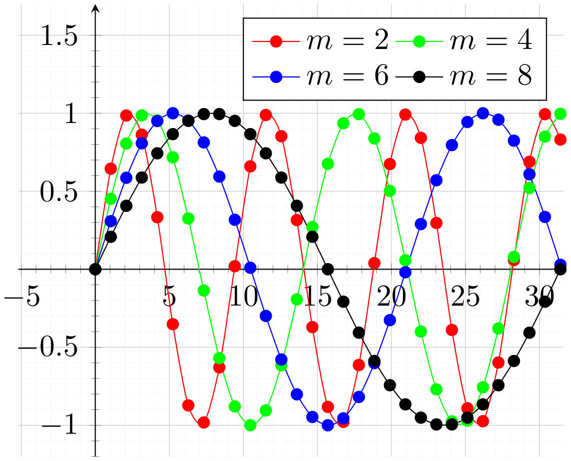

Figure 1.10 Even-index components of a sinusoidal position encoding with $M = 5, d = 8$.

mapped to a same-length sequence of vectors, one can also employ relative positions (Dai et al., 2019). That is, if each token is mapped to its own encoding given the other tokens in the sequence, the positions of the latter can be expressed relatively to the former. This will be further addressed in Section 1.3.4.

1.3.3 Token representations

The processing sequences or sets of tokens can benefit from more informed representations of these tokens. Symbolic representations of terms are hollow: terms are defined only to be different from each other, with no additional information expressing *how*. To remediate this, several algorithms have been proposed to learn dense, semantically richer token representations, termed **embeddings**. Although character embedding approaches exist (Cao & Rei, 2016), embedding algorithms are more often applied to words.

These approaches are built around distributional semantics (Boleda, 2020), the idea that words similar in meaning will occur in similar surroundings. Each word will map a dense vector representation. This representation will be refined to be able to predict co-occurring words. The manner in which the vector representation of a term is computed varies. In the case of word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), it is simply an assignment. That is, words in the vocabulary are assigned a parameter vector. In contrast, in the case of fastText (Bojanowski et al., 2017), a word embedding is computed from the vectors assigned to its character trigrams.

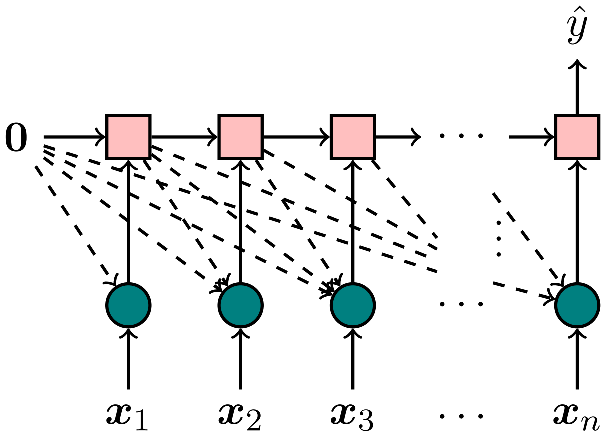

Figure 1.11 The Self-Attention mechanism applied to an RNN. Each past hidden state is matched against the current input to assess its relevance.

Word embeddings can discover complex relationships between words. Additionally, they alleviate the curse of dimension by mapping words into a smaller, space of shared components. Nonetheless, one issue with the aforementioned approaches is that the embeddings are constant and do not account for surrounding words. Thus, homographs will share a common representation. This can be addressed by contextualized representations, expounded in the following section.

1.3.4 Context

The approaches to analyzing utterances discussed so far make a single step of aggregating the representations of their tokens into a representation of the whole. Nonetheless, over recent years, there has been increased interest in approaches that revisit the tokens in light of what has been gathered about the rest of the utterance. Such approaches have largely leveraged the **attention** mechanism (Bahdanau et al., 2014), a form of weighted aggregation. At its simplest, the attention mechanism will match an attender, $\boldsymbol{a} \in \mathbb{R}^m$, against a set of n attendees, $\{\boldsymbol{x}_i\}_{i=1}^n$, $\boldsymbol{x}_i \in \mathbb{R}^m$, $i = 1, \ldots, n$. Either may be token representations or a sequence of hidden states of an RNN, for example. Each attender-attendee pair will produce an attention weight, via an attention function, $\alpha : \mathbb{R}^m \times \mathbb{R}^m \longrightarrow \mathbb{R}$, which will serve as the attention weight for that attendee. The attendees will then be aggregated together into an output, $\boldsymbol{z}$:

$$\boldsymbol{z} = \sum_{i=1}^{n} \alpha(\boldsymbol{a}, \boldsymbol{x}_i)\boldsymbol{x}_i$$

The exact nature of the attention function varies but its primary goal is to provide a weighting over the set of attendees that expresses some notion of compatibility with the attender.

Although originally introduced in machine translation as part of the decoding process— producing the translated sequence—it can also be used in encoding sequences. That is, one can let parts of the input attend over other parts. Intuitively, this evokes the notion of reassessing the significance of a token after having analyzed more of the sequence. This is termed **self-attention** (Cheng et al., 2016). In Figure 1.11, a RNN computes a new hidden state by attending over all the previous hidden states.

Taking this further, one can have each token attend over every token in the sequence, itself included, as opposed to only the past ones. This is the approach employed by **Transformers** (Vaswani et al., 2017), one of the dominating encoders in NLP in recent years. Different sets of parameters iteratively apply pairwise self-attention, allowing the updated token representation to be contextualized over several steps. Another advantage of transformers is that long-range dependencies between tokens are explicitly modelled by pairing tokens one to one.

Nevertheless, transformers have some disadvantages. One of them is their computational complexity. Whereas RNNs go over the sequence of tokens once, the complexity of pairwise self-attention is quadratic with respect to the length of token, incurring a greater computational cost with longer sequences. A strategy to circumvent this problem is to separate long sequences into segments. Crucially, to preserve some long-range dependencies between segments, the encoder is applied over a sliding window of segments (Dai et al., 2019).

Another, less apparent issue with transformers is that they operate on set, whole sequence of tokens. That is, because pairwise self-attention is predicated on matching all tokens to each other, it requires the full sequence of tokens. In contrast, RNNs can process tokens as they become available. Transformers thus leverage seemingly trivial prior information about what constitutes a sequence. Nonetheless, operating on complete sequences is seldom an issue in practice. Such prior information is easy to decide by other means (*e.g.* punctuation, white space) and sequences are usually available as wholes.

1.3.5 Language models

Language models, probability distributions over utterances, constitute an important area of research in NLP. Language models serve two main purposes in NLP. One is as pre-training for some downstream task. As described in Sections 1.1.4 and 1.2.5, this can benefit large-capacity models. Without the need for labelling, it is usually fairly convenient to gather large amounts of text to pre-train models. The second, more analytical purpose of language modelling, is to assess the ability of a model to handle the subtleties of the topology of language, to discern what is and is not said, to model language, essentially. This allows to have a working estimate of the practical computational efficiency, the data efficiency and overall aptitude of an approach in a given language.

As probability distributions, language models attempt to approximate the true distribution by minimizing the cross entropy $H(p, \hat{p})$ between the true distribution, p, and the language model, $\hat{p}$. Of course, the true distribution is not known. Nonetheless, by posing the language of choice as a stationary ergodic process[6], its distribution can be approximated to arbitrary proximity given a sufficiently large corpus (Manning & Schutze, 1999). Let $\langle x_i \rangle_{i=1}^{n}$ be the selected corpus, as a sequence of tokens. Then, the following stands:

$$H(p, \hat{p}) = \lim_{n \to \infty} -\frac{1}{n} \sum_{i=1}^{n} \log \hat{p}(\langle x_i \rangle_{i=1}^{n}).$$

Therefore, for a sufficiently large corpus, the approximation to the true cross entropy should be adequately close. This cross entropy will be expressed in bits (or nats) per token. This is easier to interpret for character-level tokenization because characters can be more readily represented with a fixed-length encoding. This is less the case for word-level tokenization. Consequently, the quality of word-level language models will usually be expressed in terms of **perplexity**:

$$\mathrm{perp}(p, \hat{p}) = b^{H(p, \hat{p})},$$

where b is the base of the logarithm used in the computation of the cross entropy. It should be noted that the stationarity assumption does not hold for any language at large because languages evolve over time. Nonetheless, a language model can be thought of as approximating a specific corpus or a language for a given time period (Manning & Schutze, 1999).

[6] A stochastic process is said to be stationary if it is invariant with respect to time shifts. It is said to be ergodic if its statistical properties can be deduced from a sufficiently long sample (see Thomas & Joy, 2006).

Naturally, the computation of the predicted probability of an utterance depends on the nature of the model. For example, a RNN will make use of the chain rule of probability:

$$p(\boldsymbol{x}_1, \dots, \boldsymbol{x}_n) = \prod_{i=1}^{n} p(\boldsymbol{x}_i | \boldsymbol{x}_1, \dots, \boldsymbol{x}_{i-1}).$$

This amounts to predicting the probability mass of the next token given the tokens parsed thus far.

Nonetheless, there are some caveats to using language modelling as a gauge for the viability of approaches. Firstly, it poses as smooth a phenomenon that is arguably not so. While human-produced utterances might show a lesser or greater *degree* of grammatical correctness in a prescriptive sense, this is not the case from a descriptive standpoint. Further, this notion might not transfer well to unintelligible token sequences. That is, there is little linguistic sense in comparing the difference in grammatical acceptability between an unintelligible sequence of token and two separate acceptable utterances. This also extends to comparing the correctness of two unintelligible sentences. Secondly, language modelling confounds together two aspects of language that linguistics treats as separate: syntax and semantics. Indeed, a language model will place grammatically unlikely and semantically unlikely but grammatically correct utterances on the same scale. Of course, a model could be explicitly conceived to treat these aspects separately or expected to learn their separation from the data. However, it remains that the language modelling framework does not account for these aspects separately.

CHAPTER 2
NATURAL LANGUAGE PROCESSING AND MENTAL HEALTH

In recent years, there has been a marked increase in research efforts towards applying NLP to **mental health**. Out of 4.9 million Canadians aged 15 and over who experienced a need for mental health care in 2015, 1.6 million felt their needs were partially met or unmet (Statistics Canada, 2017). The persistent stigma surrounding mental illness can lead to treatment avoidance or discontinuation resulting in a lack of proper expert diagnosis and support (Henderson et al., 2013).

This gap between mental health care and at-risk persons has spurred interest in new means of delivery of mental health care, including new channels for gathering signs of mental distress. For example, the vast growth in the diversity and use of online media as an outlet for personal views and experiences offers large amounts of data reflecting the day-to-day lives of its authors. Leveraging this data, however, requires to develop methods able to produce measures of clinical significance from it. Recent advances in NLP have sparked research interest in this direction.

Still, this remains challenging for several reasons. While machine learning can be leveraged to infer procedures for retrieving clinically significant variables, this requires annotated data. Data around mental health is costly to produce. Moreover, given the inherent gap between the object of study (mental status) and the object of measure (online writings), there is lack of clarity in how to select relevant portions of online behaviour to ensure the presence of relevant signals. This pushes the data collection process towards collecting potentially large amounts of data of fuzzy relevance. Downstream, this imposes large inputs on algorithms that may still require complex, computationally heavy analysis to capture potentially subtle cues about mental distress. Of course, these two constraints—heavy computation over large inputs—are fundamentally at odds. Thus, careful examination is required to strike an agreeable trade-off

The present chapter will delve into these issues as well as the ways in which they have been addressed in the literature and the remaining gaps.

2.1 Annotation

One of the difficulties in applying NLP to assist in mental health care is the annotation of data. Usual annotation difficulties aside, there is an important epistemic consideration within the problem at hand: what precisely is being annotated. When deciding the overall polarity of a film review, for example, this sentiment is attached to the review itself. In contrast, when assessing the mental status of a person *through* text, the status is attached to the person rather than the text. While this may seem a punctilious distinction, it is an important one. Indeed, evaluating the mental status of a person and diagnosing disorders especially are interactive processes wherein a clinician will probe the person for relevant information while also examining their thought process, demeanour and speech patterns (Davidshofer & Murphy, 2005). Much of this information will, in all likelihood, be missing from the text collected. Thus, as it pertains to the text, the annotation can only be thought of as indicating authorship by a person presenting a certain disorder or behaviour. Following this idea, a proper, clinically grounded annotation would consist in collecting text authored by a person whose mental status has been assessed by a clinician within the timing relevant to this assessment. This is a costly process. Scaled to the number of observations required for high-capacity models, its costs can become untenable. Consequently, research efforts exercising this approach are few. They are presented in Section 2.1.2. Instead, much work in the area has made use of related signs to provide annotations for data in order to circumvent the difficulties of obtaining clinically grounded annotations. That is, clinical truth is substituted by more accessible information to facilitate the gathering of annotated datasets. The main approaches in this practice are presented in the following section.

2.1.1 Proxy signals for annotation

Proxy signals (Ernala et al., 2019) are indicators of the pathology or behaviour studied which, while not equivalent to clinical assessment, are more readily available. This allows the collection of larger datasets than what is usually achievable with clinically grounded annotation. While some such annotation schemes will be *sound*, their validity—and the validity of ensuing predictors—will hinge on the relationship between the signals on which they rely and the underlying symptom, behaviour or pathology that they aim to reflect. That is, while these signals may be well defined and self-consistent, they may not capture the aspect of mental health that they intend. The proxy signals that have been used in the literature can be placed in four categories: Affiliation, Self-Disclosure, Expert analysis and Screening tools. They vary in their assumptions and ease of implementation as well as their limitations.

2.1.1.1 Affiliation

A first type of proxy signal are affiliations. Affiliations are associations to discussion topics or online communities explicitly made by subjects as framed by the platform in question. The precise characterization of affiliation used for study varies and is highly dependent on the nature of the platform in question. Affiliation may be subscription to specific accounts or keywords (*e.g.* hashtags) pertaining to the specific aspects of mental health in study. Alternatively, on a forum-based platform, affiliations may be determined by subscriptions to fora dedicated to specific issues of interest, that is, discussion groups wherein subjects themselves may participate.As a proxy signal, affiliations have a few advantages. They are very conveniently collected. Moreover, they are consistent in that they are defined by the platform and espoused by subjects: the signal collected is not approximate. Additionally, the use of affiliation as a proxy signal offers the possibility of collecting data from websites without subject recruitment. Nonetheless, affiliation presents severe weaknesses mainly around its validity. Indeed, a person may affiliate with outlets or discussion groups for a plethora of reasons which may not be indicative of the aspect of mental health at study. As such, affiliation is far less reliable than diagnostic mentions (Ernala et al., 2019).

2.1.1.2 Self-disclosure

Another type of proxy signal, self-disclosure, are mentions of diagnoses or clinical assessments on social media. For example, one could look for patterns such as `"I was diagnosed with` *<disorder>*`"` or `"my doctor says I have` *<disorder>*`"`. Such a statement is presumed to be more reliable than an assessment made by the individual.Then, a history of writings by the same author will be collected which will serve as the input from which predictions will be made. Naturally, this history will exclude the capturing mentions of diagnoses. Otherwise, the task of the model is reduced to finding these sequences. In addition, in most such cases, the history will precede the mention in time. Self-disclosure shares disadvantages with affiliation. Firstly, the diagnosis claims are difficult to verify, making positive annotation unreliable. Secondly, this approach also creates a significant bias in positive observation. Subjects more candid about their diagnoses might be more candid about signs and symptoms as well, reducing the external validity of the issuing findings and predictors. Finally, there is no sound strategy to collecting negative (control) observations. The inferral at play signalling a person mentioning a diagnosis as having been diagnosed does not negate well: a person who has not mentioned a diagnosis could very well have been diagnosed with the disorder at hand or a comorbid one, or could be affected by one of them without having received a proper diagnosis. This approach was introduced by Coppersmith et al. (2014), producing a dataset for depression and, later, Post-Traumatic Stress Disorder (Coppersmith et al., 2015) from Twitter. It has since been used in building datasets from Reddit[1] users for depression (Yates et al., 2017), anorexia (Losada et al., 2018) and self-harm (Losada et al., 2020).

2.1.1.3 Expert analysis of text

Another annotation scheme found in the literature is to supplement one of the two previous proxy signals (affiliation or diagnosis mentions) by human validation. That is, to have annotators validate the label found by the previously described approaches. These annotators usually have some expertise in the field. Their opinions may be informed by clinical training and experience (*e.g.* psychiatrists) or practical experience in such matters and media (*e.g.* telephone or online counselling service professionals, online support fora moderators). This strategy has the advantage that the annotation directly relates to elements found in the text. Nonetheless, a key issue with this approach is that the annotation is itself an inference, hindering not only its reliability but also its prospective precision. For example, an expert could discern whether an utterance indicates anxiety on the part of the author, but might not be able to determine the associated pathology, if any. This approach has been employed in annotating Reddit writings as indicative of suicidal ideation by both expert and crowd-sourced annotators (Shing et al., 2018).

2.1.1.4 Screening tools

A fourth, clinically sounder approach is annotating based on screening tools, such as **self-report inventories**. Self-report inventories are questionnaires surrounding specific behaviours or symptoms and completed by subjects or patients themselves (Davidshofer & Murphy, 2005). Self-report inventories need not be administered by a clinician and thus can be more widely distributed—through **crowd sourcing** for example—to gather larger amounts of data than would be possible through

[1] Reddit is a social news and discussion website. (https://www.reddit.com/)

clinician evaluation. In addition, control subjects recruited in such a manner have in fact been evaluated as per the pertinent criteria. Moreover, such inventories are much more rich in pertinent information such as signs and symptoms than binary diagnosis admission annotation. Inherent to this strategy, however, is the uncertainty associated with the self-report inventory in use. Nonetheless, overall, this approach strikes a good balance between annotation groundedness and ease of data collection. It has been used by Rude et al. (2004) in studying language patterns in essays written by depressed students. It has since been applied to the detection of depression on social media (De Choudhury et al., 2013; Losada et al., 2020).

2.1.2 Clinically grounded annotation

Given the uncertainties of these different types of proxy signals—particularly in regard to their validity, it may be preferable to rely on annotation procedures that are closer to clinical truth. That is, in order to fully understand how NLP approaches may relate to clinical realities, the annotations on which they lean should reflect these realities. These clinically grounded annotations are more costly to produce and are therefore rare in the literature. Notable examples are Ernala et al. (2019), who linked subject Twitter[2] and Facebook[3] data to schizophrenia diagnoses, and Merchant et al. (2019), who associated Facebook data to past diagnoses of depression, anxiety, psychoses, among other non-mental conditions (Eichstaedt et al., 2018).

2.2 Challenges in mental health assessment from text

Leaving aside annotation challenges, mental health assessment is a difficult family of tasks from a purely technical NLP standpoint. While much of this difficulty extends to NLP practice at large, there exist three key concerns that are much more specific to mental health assessment from text. These concerns are at the heart of the work presented in this thesis. These are granularity, actionability and temporality. They are expounded in the following sections.

[2] Twitter is a micro-blogging platform. (https://twitter.com/)

[3] Facebook is a social network. (https://facebook.com/)

2.2.1 Granularity and aggregation

Cues about the issues that the language of at-risk persons may intimate are subtle. Reliance on low-level, internal features (bag-of-words from extracted vocabulary) is therefore not always effective (Merchant et al., 2019; Yates et al., 2017). Nonetheless, integrating low-level, external features, appears to help in this regard (De Choudhury et al., 2013; Ernala et al., 2019). Concurrently, word-sequence approaches show promise (Ive et al., 2018; Shing et al., 2020; Yates et al., 2017). Of course, injecting prior knowledge in a structured, high-level manner is not trivial: it requires some working notion of how the linguistic components, as they pertain to mental health, might interact with each other. The alternative is then to rely on high-capacity models to infer these relations. These approaches have utilized RNNs (Ive et al., 2018; Yates et al., 2017), CNNs (Shing et al., 2018) as well as transformer (Madani et al., 2020; Martínez-Castaño et al., 2020) and related approaches (Maupomé et al., 2020).

Because annotation is provided at the subject level, supplemental issues arise. Whereas the vast majority of classification and regression tasks in NLP are predicated on relatively short and consistent documents, assessing mental status from textual content from social media requires the aggregation of numerous documents. These documents may not only constitute a large amount of text but also vary greatly in subject matter, register and tone. In bag-of-words approaches, the aggregation of writings is immediate. The entire history can be represented as a single document (Armstrong et al., 2021; Maupomé et al., 2021a).

In term-sequence approaches, aggregation is more complicated. Typically, the writing representation is learned in conjunction with the aggregation and the prediction in place in a back-propagation framework. The writing representation must therefore be applied to all the writings in a history at once. Alternatively, in training, writings can be sampled from the history, acting as a regularizer (Maupomé et al., 2019, 2020). This alleviates the computational burden in training. Nonetheless, an issue with this approach is that prediction must act on the entire history. This shift from small, fixed-size samples to larger, variable-size histories can cause the predictors to be destabilized. Approaches to back-propagated aggregation can be simple, such as averaging the vector representations of writings (Martínez-Castaño et al., 2020; Yates et al., 2017). This same idea can also be supplemented by attention (Maupomé et al., 2020). Alternatively, the order of writings can be taken into account by feeding the writing representations to an RNN (Ambalavanan et al., 2019;

Ive et al., 2018). Attention-based aggregation can also be performed in a flat manner, *i.e.* allowing the encoder to attend over portions of each writing (Maupomé et al., 2019, 2020).

2.2.2 Actionability

Beyond the reliability of prediction, an important consideration is how these predictions can help the diagnostic and treatment processes. In AI at large, there is growing interest in interpretable models, mainly as it pertains to transparency and fairness. In applying NLP to mental health, this is also considered a matter of informativeness. An approach that has garnered much interest is to interrogate attention weights post-hoc in order to find the portions of an observation that influenced most the predictor output (Ive et al., 2018; Shing et al., 2018). This can be applied both at the inter-writing level, to determine the most salient writings, and within each writing to determine relevant excerpts. This is a sensible approach due to the intuitiveness and efficacy of attention mechanisms. However, recent work shows there is no strict correspondence between particular weight values and outputs, calling to question their explanatory power (Grimsley et al., 2020). More importantly, there is little reason to expect the salient excerpts or writings to pertain to the clinically recognized signs of the disorder or harmful behaviour at hand.

For this reason, the informativeness of predictor outputs, should be construed as a matter of *actionability* (Maupomé et al., 2021a). That is, for the deployment of machine learning predictors to serve the diagnostic and treatment processes, their predictions should be structured in a clinically meaningful manner. To this end, the annotation should contain a structured assessment such as those provided by self-report inventories.

2.2.3 Temporality

Mental health and its possible concerns evolve over time. Consequently, these aspects should be addressed by NLP applications. There are two facets to temporality in mental health assessment from text: the temporal offset in assessment and the temporal modelling of the textual manifestation of mental health concerns. Both of these facets have been addressed in the literature.

Beyond increased accessibility, automated assessment of mental health from text could also allow for the possibility of earlier intervention in the case of risks related to mental health. Generally, annotation usually does not provide ground-truth markers of when the illness or harmful behaviour started to manifest in the text, the notion of delay in detection is understood as the overall span of time needed by a predictor to provide a decision. Nonetheless, the writings are usually collected in a time window determined by the annotation. For example, in the case of diagnosis mentions, the time window will be set to end before the earliest mention. In the case of self-report inventory or clinician assessments, the time window will end before it. Additionally, because these approaches are not interactive, there is no means of ensuring a certain rate of textual production from subjects—or indeed uniformity in rate of production between subjects. Consequently, the delay in decision from predictors is sometimes measured in terms of textual production rather than wall-clock time. This can be measured in number of writings or tokens.

Assessing the timeliness of assessments has been mostly evaluated with respect to positive decisions regarding a threat to mental health. Losada et al. (2018) propose an early risk decision error measure for binary classification which counts the number of writings processed by the predictor. True positives are weighted by applying a logistic function on the delay in decision with respect to a set threshold. False negatives are counted as infinite true positives. Shing et al. (2020) argue the problem should be approached as one of nested risk prioritization: subjects should be ranked in decreasing order of risk as should the writings of each subject. The reasoning behind is that, given finite human resources for review, reviewers should be given the most urgent subjects first and the most concerning writers for each subject.

As for the temporal awareness in models, the most common approach is to treat writings as a sequence (Ive et al., 2018; Shing et al., 2018, 2020). While this accounts for the order of writings, it does not account for the time elapsed between them. This can be addressed by incorporating the time elapsed with respect to the latest writing (in days, for example) as their position (Maupomé et al., 2020). A third and final aspect of the temporality of writings is the time of day of their publication, as well as the time of the week (Altszyler et al., 2018).

Together with annotation difficulties, the peculiarities of mental health analysis from text expounded in this section make it a challenging and distinct problem in NLP. It is apparent that the sought-after predictor, capable of providing detailed analyses, must gather signals that are subtle and dispersed through large amounts of text and whose relation to the phenomena they manifest are not well understood. It has been the goal of the work presented in this thesis to tackle these issues. The particular components of this work are the subject of the following chapters.

CHAPTER 3
LEVERAGING TEXTUAL SIMILARITY TO PREDICT BECK DEPRESSION INVENTORY ANSWERS

3.1 Preface

In 2020, Losada et al. (2020) put forth a dataset aimed at studying the viability of establishing a tally of depression symptoms from social media text. To this end, several subjects were asked to fill out a standard self-report questionnaire, the **Beck Depression Inventory-2nd Edition (BDI)** (Beck et al., 1996), and to grant access to their **history** of writings on the social media platform, Reddit. The BDI is a standard self-report tool covering 21 signs and symptoms of depression. Thus it provides large coverage of facets of the pathology that may emerge over a vast span of writings.

In machine learning terms, however, this dataset presents some of the challenges introduced in Chapter 2. Namely, it contains a limited number of examples, each spanning a large history of writings. The larger number of targets compared to a binary classification task raises a few issues. While the increased complexity in output gives rise to sparsity of data in the output space—the curse of dimension—it provides more feedback signals in training. However, this is difficult to exploit in practice. One manner of addressing this complex prediction with few examples would be to separate prediction to each BDI item independently. Moreover, these predictions could be restricted to analyze smaller segments of the histories selected through some other prior criterion. However, such compartmentalization of predictions would have two flaws: foregoing the dependencies that exist between items, and being bounded by the assumptions of the prior segment selection.

Instead, the work presented in this chapter made use of an unsupervised setting to infer representations of writing history. Such representations are then used for the BDI prediction task. Of course, this requires a choice of unsupervised task that elicits useful representations for depression symptom assessment. Likewise, the model should be able to produce such a representation and do so efficiently from large amounts of text.

To this end, the present work proposed authorship verification as such a task, with **Deep Averaging Networks (DANs)** as the model to deploy towards it. Broadly, authorship verification seeks decide whether two excerpts were authored by the same person. Thus, assuming that persons writing similarly may present similar symptoms of depression, representations that can discern authorship will contain information that will able to compare authors along depression signs and symptoms. That is, in an "authorship space", persons presenting similar depression symptoms may appear closely together. The reason behind this conjecture is that, while authorship objectives may not explicitly elicit psychosocial factors from text, they do, at least in their intent, favour aspects of depression that pertain to the *author* rather than the subject matter of the text. Training only requires data with text and author identification. To this end, data from a previous, related eRisk set was used. This dataset concerns binary risk assessment of depression (Losada et al., 2018). However, risk labels were not used.

This is done through the use of DANs, a neural network approach to text representation that operates by averaging together vector representations of tokens and feeding them through a FFN. While not built with explicit considerations of authorship, DANs are a flexible approach to representation learning that scale well to large amounts of text. Indeed, their computational complexity is linear with respect to the length of the sequence. Moreover, the averaging together of vector tokens stably allows for changes in their number. As such, they allow for the encoding of large and variable amounts of text.

Thus, the present work seeks to evaluate the use of DANs as authorship verification functions, and of the features they elicit, as a representation space wherein neighbouring subjects may present similar depression symptoms, as described by the BDI.

3.2 References

Diego Maupomé, Maxime D. Armstrong, Fanny Rancourt, Marie-Jean Meurs (2021). Leveraging Textual Similarity to Predict Beck Depression Inventory Answers. *Proceedings of the* 34th *Canadian Conference on Artificial Intelligence. (pp.1-12).* Publisher: Canadian Artificial Intelligence Association. `https://doi.org/10.21428/594757db.5c753c3d`

Open-access publication: this article is licensed under a Creative Commons Attribution International license (CC BY 4.0).

Open-source code: the source code of the proposed systems is available here `https://gitlab.labikb.ca/ikb-lab/nlp/canadian-ai-2021/textual-similarity` and is licensed under the GNU GPLv3.

3.3 Publication

Leveraging Textual Similarity to Predict Beck Depression Inventory Answers[1]

Diego Maupomé, Maxime D. Armstrong, Fanny Rancourt, Marie-Jean Meurs

Université du Québec à Montréal, Montréal, QC, Canada

3.3.1 Abstract

This work proposes an approach to predict potential answers to the BDI, a 21-item self-report inventory measuring the severity of depression in adolescents and adults. Predictions are based on similarity measures between the textual productions of social media users and completed BDIs. Two methods of establishing similarity are compared. The first one is using unsupervised extraction of topics, and the second one is based on authorship attribution through the use of neural encoders. Both approaches achieve interesting results, indicating that the authorship attribution task can induce a similarity measure useful for depression symptom detection. The issues that arise in predicting several aspects of depression are further discussed.

3.3.2 Introduction

Depression is a widespread mental disorder, affecting more than 264 million people of all ages worldwide (James et al., 2018). The wide spectrum of mental disorders accounts for 13% of the total global burden of disease, with depression alone representing 4.3%. People with such disorders have disproportionately higher rates of disability and mortality, with up to 40-60% more chance to die prematurely than the general population (World Health Organization, 2013). In addition to the threat that mental disorders poses to public health, the economic impact of these is major, resulting in an estimated global economic lost output of US$ 16.3 trillion between 2011 and 2030 (Bloom

[1] The layout of this article has been modified to match the format of this thesis

et al., 2012).

In the United States, around 17.3 million adults had experienced at least one major depressive episode in 2017, representing 7.1% of all U.S. adult population (Substance Abuse and Mental Health Services Administration, 2018). In Canada, which is among the few countries where very detailed statistics are available, 20% of the population will personally experience a mental health issue at any given year. In particular, approximately 8% of adults will experience major depression at some time in their lives (Canadian Mental Health Association, 2020). A recent study by Qin et al. (2018) shows that the prevalence rate of depression among the adult population in China is estimated as high as 38%.

Despite the proportion of the population affected by mental health issues, and the overall repercussions on society, finding adequate help and resources is burdensome for those who need it, especially for young people. In 2018, up to 76% of the care receivers aged 15 to 34 in Canada affirmed that, were it not for their family and friends, they would have experienced difficulty in finding help (Statistics Canada, 2020). Moreover, the stigma surrounding mental illness can lead to treatment avoidance, delays to care, and discontinuation of treatment, leading to a lack of proper expert diagnosis and support (Henderson et al., 2013).

For these reasons, there is burgeoning interest in utilizing automated means to help bridge the gap between at-risk persons and mental health services. In particular, in Computational Linguistics, the use of Internet fora and social media to discuss these matters offers an opportunity to devise methods of mental status assessment from text. As such, there are increasing research efforts towards risk detection of mental health problems and self-harming behaviors (Shing et al., 2018; Coppersmith et al., 2015; Maupomé et al., 2019).

However, for these analyses to be fully useful to the diagnostic and treatment processes, the predictions made must be as informative as possible. This can be broadly construed as a matter of interpretability. In doing so, recent work has been aimed in this direction (Ive et al., 2018; Shing et al., 2020). Nonetheless, we contend that the primary concern should be actionability. That is, the predictions should contain information that is clinically meaningful, *i.e.*, pertaining to symptoms.

Thus, the work presented focuses on the dataset put forth by Losada et al. (2020). This corpus is composed of textual content written on Reddit by 90 social media users, paired with a standard questionnaire covering signs and symptoms of depression, the **Beck Depression Inventory-2nd Edition (BDI)** (Beck et al., 1996). While detailed, this dataset is of modest size, making it difficult to learn the mapping from text to BDI answers directly. Instead, the prediction of answers is based on those of known persons in a nearest-neighbor fashion.

The main contributions of this work are:

1. Comparing the effectiveness in predicting BDI answers of two methods of establishing similarity. The first is based on the unsupervised extraction of topics from **textual productions**. The second consists of explicitly learning similarity as a matter of authorship attribution through the use of neural encoders.
2. Measuring whether the dependencies between different aspects of depression (as per the BDI) can be leveraged in such a framework.

The paper is organized as follows. Section 3.3.3 presents the prediction algorithm as well as the similarity measures used, and similar work in the literature. Section 3.3.4 describes the evaluation metrics, the experimental settings as well as the results obtained. Finally, Section 3.3.5 discusses our findings and Section 3.3.6 concludes this paper.

3.3.3 Resources and Methodology

3.3.3.1 Dataset and BDI-II structure

As previously mentioned, the dataset proposed by Losada et al. (2020) is composed of the textual production 90 Reddit users, each one being paired with BDI answers. The number of writings per person ranges from 12 to 1164 (median of 234). These writings range in word count from 11 to 1515 (median of 34).

The BDI is a 21-item self-report measure of depressive symptoms experienced during the past week (see questions and possible answers provided in the appendix). It covers different aspects of the manifestation of depression, related to affective, cognitive, somatic and vegetative symptoms, according to the DSM-IV (Bell, 1994) criteria for major depression[2]. Each item is associated with a single symptom allowing self-assessment of a specific behavioral manifestation of depression. Items are ranked to reflect the range of severity of the symptom from 0 (absent) to 3 (severe). The BDI-II covers symptoms range from the very concrete, such as changes in sleep pattern and appetite, to more abstract aspects, such as guilt or punishment feelings. Depression severity is scored from the BDI-II by adding the ratings for all the 21 items, obtained from self-assessment. The minimum score is 0 and maximum score is 63, with higher scores indicating greater symptom severity. Depression levels are divided in 4 categories: minimal (scores of 0–13), mild (14–19), moderate (20–28) and severe (29–63).

The BDI was filled out by the 90 participants immediately before the collection of their textual productions. Two key difficulties of this dataset are the limited amount of annotated observations as well as their large size in terms of textual content. However, to our knowledge, this dataset is the first containing a clinically meaningful, detailed inventory of signs of depression.

3.3.3.2 Similar textual production, similar potential symptoms?

With a sound measure of similarity between the textual production of different persons, one could predict a new person's answers to the BDI based on those of a *pool* of known persons, using the textual production similarity to weight their respective importance. Let $\alpha(u, v) \in [0, 1]$ be such a similarity between the textual productions of persons u and v, with a greater number denoting greater similarity. Let A be the set of answers to some question in the BDI, with $\mathbf{1}_a(v)$ indicating whether person v selected answer $a \in A$. The predicted answer among A for person u given a pool of persons V is given by:

$$\underset{a \in A}{\operatorname{argmax}} \sum_{v \in V} \alpha(u, v) \mathbf{1}_a(v)$$

In this way, there is no need to learn the map from persons' textual productions to BDI answers but one can simply exploit the hypothesis that persons producing similar contents (as per the similarity measure) could present similar potential symptoms. This approach still allows for these persons to diverge in some answers, provided that others, less similar in their production, collectively agree, as illustrated in Fig 3.1. Furthermore, the answer prediction can be made based solely on a fixed

[2] The American Psychiatry Association (2013) reports that the core criterion symptoms applied to the diagnosis of a major depressive episode has not changed from DSM-IV to DSM-V, making the BDI-II valid for both versions.

	v_1	v_2	v_3	v_4	
$\alpha(v_i, u)$	0.4	0.2	0.1	0.3	u
`Q1`	`0`	`0`	`3`	`1`	→ `0`
`Q2`	`0`	`1`	`2`	`1`	→ `1`
`Q3`	`1`	`0`	`2`	`3`	→ `1`
⋮		⋮			⋮

Figure 3.1 Prediction example on BDI answers from person u based on the similarity $\alpha(v_i, u)$ between the textual productions of persons $v_i : i = 1, \ldots, 4$ and u. Lines 2 to 4 indicate values of the BDI answers given by persons v_i to questions `Q1` to `Q3`. At each question, the answer with highest total score is selected.

number of nearest neighbors, or on the entire pool of persons.

However, no such similarity measure is available off-the-shelf. This work hence attempts to induce it considering two methods. The first one consists in extracting topics from textual productions, then compare persons' textual productions in the topic space. The second method approaches the task of learning similarity between persons' textual productions as an authorship attribution task. That is, train a model to decide whether two sets of textual productions were written by the same person or not.

3.3.3.3 Topic extraction

As the first method is based on unsupervised topic extraction, a reliable and robust approach is required to capture the underlying topic structure from depression related textual productions. Latent Dirichlet Allocation (LDA) (Blei et al., 2003) was selected since it has been shown to be meaningful to depression symptomatology (Coppersmith et al., 2015; Resnik et al., 2015; Maupomé & Meurs, 2018). LDA allows the inference of topics (distributions over tokens) in an unsupervised manner. Once the topics are extracted from the concatenation of each person's textual productions, the textual productions can be represented as vectors of topics, which are then used to compute their similarity. Similarly, TF-IDF is used as a baseline (Wu et al., 2008).

3.3.3.4 Authorship attribution

As mentioned, the textual productions of persons in the dataset can be large both in number of documents and in length of documents. Moreover, they vary in both of these respects. For these reasons, we require an encoding architecture that is computationally efficient and invariant to the quantity of text encoded. Therefore, DANs (Iyyer et al., 2015) are considered for encoding textual productions. DANs build a dense encoding of a document by averaging together the vector representations of the tokens making up the document and feeding it through a deep feed-forward network. Here, tokens are represented as one-hot vectors. The average is then equivalent to a token-frequency representation of documents. This allows for efficient pre-processing of documents and a normalized representation, which would be less sensitive to document or word count. The DAN thus can transform the entirety of a person's textual production into a vector representation used to compute the similarity.

The encoder itself is trained in an unsupervised manner. Pairs of persons are drawn at random with replacement, *i.e.* the pair may contain the same person twice or different persons. The encoder must decide whether the associated TPs were written by the same person or not. To avoid overfitting, regularization in the form of dropout is applied, both in documents and in length of documents. Specifically, a small set of each person's textual production is sampled to feed to the encoder. Crucially, a new sample is taken at every training step. Similarly, word dropout (Iyyer et al., 2015) is applied on the tokens in the bag-of-words representation. In validation, however, the entire textual production is considered.

3.3.3.5 Alternative approaches

There has been substantial work in predicting the risk for mental health issues or self-harming behaviors from text. Nonetheless, these predictions are usually of *degree* of risk, either binary (Losada et al., 2018), or with several levels (Zirikly et al., 2019). Some recent efforts have focused on attempting to signal the portions of text that most contribute to the prediction, thus potentially offering some details on the manifestation of the issue.

The current state of the art for a variety of natural language processing challenges (see Wang et al., 2019a) has been dominated by Transformer-based approaches (Vaswani et al., 2017). Nonetheless, such approaches are ill-fitted here. They can be computationally intensive and offer no clear manner of aggregating groups of documents. In contrast, DANs can be applied directly to token-count representations, greatly reducing processing time.

As for prediction of BDI answers, the eRisk 2019 and 2020 shared tasks resulted in a growing interest from the research community (Trifan & Oliveira, 2019; Oliveira, 2020; Maupomé et al., 2020; Uban & Rosso, 2020; Martínez-Castaño et al., 2020). Some of these involve direct learning of textual psycho-linguistic features relevant to depression (Oliveira, 2020; Uban & Rosso, 2020). Other direct approaches involve the use of pre-trained Transformers (Martínez-Castaño et al., 2020). To address the computational difficulty of handling long textual productions, the authors propose truncating or splitting documents into more manageable lengths. Nonetheless, this presupposes that any portion of a person's textual production is pertinent to their depression symptoms or lack thereof. An approach comparable to this work, taken by (Abed-Esfahani et al., 2019) sought to compare the textual output of persons and the text contained in the BDI questionnaire. However, the reported results are modest, mainly due to the brevity of the questionnaire text.

3.3.4 Experiments

3.3.4.1 Metrics

Losada et al. (2020) propose four metrics to evaluate the prediction of BDI answers. As mentioned, the questionnaire is composed of 21 questions with 4 possible answers (from 0 to 3), except for questions 16 and 18, where there are seven possible answers (0, 1a, 1b, 2a, 2b, 3a, 3b). The metrics, described hereafter, aim to evaluate exact correspondence, per-item closeness and overall closeness.

- The **Average Hit Rate (AHR)** measures the proportion of exactly correct predictions. If the hit rate for a given person is defined as the proportion of questions for which the prediction matched exactly the true answer, the AHR is the mean hit rate across all persons.
- The **Average Closeness Rate (ACR)**, by contrast, measures how close the predicted answers were to the true answer. The absolute differences between the value of the predicted and true answers are computed. For questions 16 and 18, each pair (1a,1b), (2a,2b), (3a,3b) is considered to have the same value. Then, the absolute difference, ad, is compared to the maximum possible absolute difference, 3, to give the closeness rate, $CR = \frac{3-ad}{3}$. The closeness rate for a person is

the mean closeness rate across questions. The ACR is then simply the mean closeness rate across persons.

- The **Average Difference between Overall Depression Levels (ADODL)** measures the difference in total score. As previously mentioned, the answers to each question in the BDI are summed up to compute an overall depression level. The Difference between Overall Depression Levels (DODL) is simply the absolute difference between this total as per the ground truth and as per the predicted answers, normalized, $DODL = \frac{63-ad}{63}$, 63 being the maximum possible depression level. As with the previous metrics, the ADODL is the mean DODL over all persons. As previously mentioned, depression levels as measured by the BDI can be assigned to one of four categories.

- The **Depression Category Hit Rate (DCHR)**, is the accuracy of the categorization afforded by the predicted BDI.

3.3.4.2 Similarity learning

All models are trained on data collected by Losada et al. (2018). The dataset consists of textual productions on Reddit fora from 1707 persons, who are categorized as either at risk of depression or not. For each approach, two forms of tokenization are used in separate models: in-word character trigrams and word stemming. Tokens used by fewer than 25 persons are excluded. The labels are ignored.

The inverse document frequencies and LDA topics are computed using each person's textual production as a single document. The LDA model was built with Gensim, trained over 50 iterations with the Variational Bayes method. The number of topics was empirically derived. The best results are often acquired when performing LDA around 50 topics in a depression-related context (Resnik et al., 2015). For this reason, several models with varying numbers of topics were trained, to discover that using LDA with 30 topics on this specific dataset gave the leading performance. Training both models (trigrams and stems) took about 90 minutes on a 2.8 GHz Intel Core i5 CPU with 8GB of memory.

As for DANs, the training considerations are as follows. Firstly, note that the number of positive unordered pairs of TPs (same person) grows linearly with the number of persons, while the number

Table 3.1 Results of the proposed models on BDI prediction with persons split into training and test as per Losada et al. (2020). The BDI predictions were tested considering the $k = \{3, 5, 7, 9, -\}$ nearest neighbors ('-' for all). The results are the best for each metric attained by the model, with the corresponding k in parentheses. The bottom of the Table presents the best results obtained at eRisk 2020 for each metrics.

Model	Tokenization	AHR	ACR	ADODL	DCHR
TF-IDF	trigrams	**.396** (9)	**.701** (3)	.812 (-)	.333 (-)
TF-IDF	stems	.391 (9)	.697 (3)	.810 (-)	.348 (9)
LDA	trigrams	.364 (-)	.689 (-)	.815 (-)	.261 (5)
LDA	stems	.380 (-)	.696 (-)	**.832** (-)	**.420** (9)
DAN (small)	trigrams	.386 (-)	.698 (-)	.823 (5)	.333 (9)
DAN (small)	stems	.364 (-)	.689 (-)	.813 (-)	.348 (5)
DAN (large)	trigrams	.380 (-)	.694 (-)	.820 (5)	.333 (3)
DAN (large)	stems	.374 (-)	.695 (-)	.824 (-)	.319 (3)
Zeroes		.364	.644	.644	.145
Ones		.310	.735	.819	.246
Random		.232	.586	.753	.271
BioInfo@UAVR (Oliveira, 2020)		**.383**	.692	.760	.300
iLab (Martínez-Castaño et al., 2020)		.371	**.694**	.817	.271
prhlt (Uban & Rosso, 2020)		.346	.674	.806	**.357**
relai (Maupomé et al., 2020)		.364	.683	**.832**	.343

of negative ones (different persons) grows quadratically. There is no prior reason to favor positive or negative pairs in training the encoder. Therefore, the negative class is undersampled in training, generating positive and negative pairs in equal amounts. In validation, however, it is relevant to reduce the variance brought on by validating on any particular set of negative pairs. Therefore, all possible unordered pairs are generated, validating on the same set of pairs at each epoch. Thus, the balanced accuracy is monitored on a 10% validation set in order to perform early stopping (Kelleher et al., 2020). This is computed as the unweighted average recall of each of the two classes.

Secondly, to avoid overfitting, in training, only a sample of 20 documents is taken from each person's textual production. Persons having produced fewer than 50 documents are excluded. When matching a person's textual production against itself (positive pair), the samples are mutually exclusive. Moreover, word dropout (.15) is applied on the tokens, independently so across textual production samples. The encoders are trained by gradient descent, using the Adam algorithm (Kingma & Ba, 2014) over 50 epochs and performing early stopping as per the balanced accuracy on the validation set. The models were built with Tensorflow. One large and one small model were trained for each tokenization scheme. Large models consist of 9 layers of 1024 units whereas small ones consist of 9 layers of 512 units. This results in large models with 25.1M and 23.1M parameters for stems and trigrams, respectively, and small models of 10.4M and 9.4M parameters, likewise respectively. All layers, including the final one, use rectified linear unit activation. All models were initialized by He normal initialization (He et al., 2015) and trained with mini-batches of size 32. The total running time for all four models was about 3 hours on a 3.6 GHz Intel Core i7-7700 CPU with 31.2GB of memory. The balanced accuracies achieved in validation were .836 for the small trigram model, .859 for the small stem model, .852 for the large trigram model and .834 for the large stem model.

Table 3.2 Comparing the results of the proposed models on the BDI prediction in 5-fold cross validation and authorship decision tasks. The BDI predictions were tested considering the $k = \{3, 5, 7, 9, -\}$ nearest neighbors ('-' for all). The results are the best for each metric attained by the model, with the corresponding k in parentheses. For authorship, in addition to the accuracy, we report the average similarity score produced by the model for negative and positive pairs of documents. For reference, the results of predicting only `0`, only `1` or a uniformly random answer to each question are provided.

		BDI				Authorship		
Model	Tokenization	AHR	ACR	ADODL	DCHR	pos sim.	neg sim.	acc
TF-IDF	trigrams	.378 (-)	.697 (7)	.805 (3)	.311 (3)	.906 ±.132	.852±.126	.509
TF-IDF	stems	.381 (-)	.670 (9)	.807 (3)	.278 (9)	.825±.193	.821±.164	.495
LDA	trigrams	.387 (-)	.698 (-)	.794 (9)	.289 (3)	.763±.285	.361±.280	.760
LDA	stems	.386 (-)	.670 (-)	.799 (-)	.322 (3)	.863±.204	.446±.291	.761
DAN (small)	trigrams	**.406** (-)	**.713** (-)	**.824** (7)	**.388** (7)	.878±.188	.445±.301	.746
DAN (small)	stems	.396 (-)	.709 (-)	.816 (5)	.378 (5)	.849±.211	.346±.284	**.804**
DAN (large)	trigrams	.396 (-)	.710 (-)	.797 (7)	.356 (3)	.911±.148	.567±.323	.678
DAN (large)	stems	.403 (-)	**.714** (-)	.804 (9)	.310 (9)	.852±.205	.378±.297	.788
Zeroes		.359	.636	.636	.156			
Ones		.302	.730	.814	.244			
Random		.229	.584	.758	.284			

3.3.4.3 Prediction of BDI-II answers

Once the encoders are trained, the BDI answers of persons is predicted based on similarity to a subset of the textual production of persons whose BDI is known. The similarity itself is computed as the cosine similarity between the encoded textual productions. These encodings being positive, the cosine similarity is in the unit interval. The dataset, as put forth by Losada et al. (2020), is split into the textual productions of 20 persons for training and the textual productions of 70 persons for test. However, the distribution of BDI scores are quite different between these two subsets: the number of persons in each of the four depression categories are (4, 4, 4, 8) for the training set and (10, 23, 18, 19) for the test set. Therefore, we report results on the original split as well as those obtained by stratified 5-fold cross validation. The experiments are conducted basing prediction on the nearest 3, 5, 7 and 9 neighbors as well as the entire pool for prediction. These results are compared to those obtained by answering only `0` or only `1` (the most frequent answers) to each question, as well as uniform random prediction averaged over 100 iterations.

Table 3.1 presents the results in prediction of BDI answers using the original training-test split, including the best results obtained at eRisk 2020 for the sake of comparison. Table 3.2 presents the results on cross-validated BDI prediction. Furthermore, Table 3.2 shows the results of our different models on authorship decision, notwithstanding that the TF-IDF and LDA models were not trained for this task. These results are computed on the entire dataset (textual productions of 90 persons). Negative pairs are generated using the entire textual production of each person. Positive pairs are generated by partitioning each person's textual production in half. These partitions are resampled enough times to arrive at roughly equal numbers of positive and negative pairs.

Table 3.3 Mode of the answers to each question in the dataset (90 users).

question	Q1	Q2	Q3	Q4	Q5	Q6	Q7
mode	1	1	1	1	0	0	0

question	Q8	Q9	Q10	Q11	Q12	Q13	Q14
mode	0	0	0	0	1	0	0

question	Q15	Q16	Q17	Q18	Q19	Q20	Q21
mode	1	1b	0	0	0	1	0

3.3.5 Discussion

The proposed models achieve interesting results both in cross validation (Table 3.2) and on the original split (Table 3.1), indicating that the authorship decision task can induce a similarity measure useful for depression symptom detection. As shown in Table 3.1, these models achieve results comparable to those reported by Losada et al. (2020) on the original split, with the best model surpassing the previous best DCHR. While authorship seems a good proxy for predicting BDI answers, the converse does not hold, as the TF-IDF model performs quite poorly in authorship attribution, as shown in Table 3.2. Upon closer examination, on the cross-validation experiments, TF-IDF models predict always the same BDI answers when considering a large number of neighbors. This is not the case for the other models.

Table 3.4 Results of the proposed models on predicting BDI answers in five fold cross validation using DkNN and MkNN. The predictions of BDI answers were tested considering the $(k, \delta) \in \{(5,3),(7,5),(9,5),(20,10),(30,10),(40,10)\}$ for DkNN and $k \in \{5,7,9,20,30,40\}$ for MkNN. The smoothing parameter for DkNN, s, is set to 1. The results are the best for each metric attained by the model, with the corresponding value(s) of k in parentheses.

		MkNN				DkNN			
Model	Tokenization	AHR	ACR	ADODL	DCHR	AHR	ACR	ADODL	DCHR
TF-IDF	trigrams	.369 (40)	.691(40)	.795(40)	.306(30)	.368 (40)	.691(40)	.795(40)	.306(30)
TF-IDF	stems	.367(30)	.697(40)	.809(40)	.306(5)	.367(30)	.699(40)	.809(40)	.292(20;5)
LDA	trigrams	.361(9)	.690(9)	**.829**(40)	.361(40)	.360(9)	.690(9)	**.830**(40)	.361(40)
LDA	stems	.343(30)	.678(40)	.818,(40)	**.431**(40)	.343(30)	.678(30)	.817(40)	**.417**(40)
DAN (small)	trigrams	**.374**(30)	**.707**(40)	.820(30)	.347(9)	**.374**(30)	**.707**(30)	.821(9)	.347(9)
DAN (small)	stems	.362(20)	.701(30)	.823(30)	.333(5)	.362(20)	.701(30)	.823(30)	.333(5)
DAN (large)	trigrams	.358(9)	.698(30)	.827(30)	.347(30;9)	.355(30)	.698(30)	.827(30)	.347(30)
DAN (large)	stems	.372(20)	.705(30)	.818(40)	.306(40)	.372(20)	.705(30)	.818(40)	.333(9)

The BDI answers predicted by TF-IDF are very close to the mode of the answers to each question in the dataset of 90 users, reported in Table 3.3. As a consequence, TF-IDF performs well on average, especially on distance-based metrics, such as ACR and ADODL. This is reflected by the proximity in ACR and ADODL of predicting `1` to all questions, which is also close to the mode of the answers to each question. On the original split, where the pool of known persons is limited, the BDI answers predicted by the TF-IDF vary. This sensitivity to the pool of known persons is reflected by the similarity computed in the authorship attribution task. As shown in Table 3.2, TF-IDF models produce the smallest range of similarity scores when shown excerpts from different

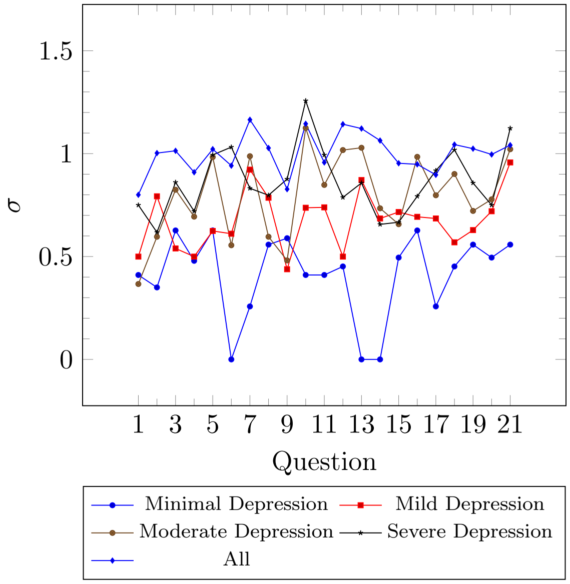

Figure 3.2 Standard deviation in answers to the BDI among persons in each depression category

persons. Obviously, the authorship attribution task is a proxy. By the very premise of the algorithm predicting BDI answers, the textual productions compared never issue from the same person. What is interesting is the expressiveness of the similarity function, which requires, at a minimum, a wide range of values. This explains why, with all persons in the pool within a small range of similarity, numbers win out, and the answers predicted are the most common ones. It also points to a clear, inherent limitation of the approach at large. Collapsing the similarity into a single scalar expresses only *if* two persons produce similar content and not *how*. The aspects of similarity are instead expected to appear by comparing to the BDI answers of different persons.

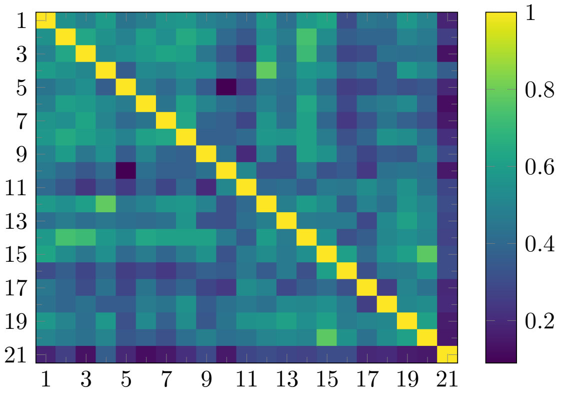

Figure 3.3 Pearson correlation coefficient between questions on the BDI among the 90 persons in the dataset

The more basic issue with this premise is that, while the computed similarity between persons' textual productions remains the same, their similarities as per the BDI answers vary in many ways. Firstly, the answers are differently spread out for each question. The standard deviation for each question over all filled BDI ranges from 0.8 for question 1 to 1.16 for question 7. Secondly, the variation changes with overall score. This is illustrated in Figure 3.2. When grouping persons by depression category, the spread on each question varies between categories. For example, answers from persons in the minimal depression category present a null standard deviation for questions 6, 13 and 14. On the other hand, answers to questions 10 and 21 present a standard deviation among the moderately and severely depressed close to that of a uniform distribution ($\sqrt{5}/2$). Moreover, there are dependencies present between questions in the BDI, as evidenced by the heatmap in Figure 3.3.

For these reasons, further experiments were conducted with the Dependent Multi-label k-Nearest Neighbour () (Younes et al., 2011) algorithm, using the same similarities between textual productions already computed. This algorithm verifies the agreement in other variables among the neighbors of the test instances as well as those of the pool of known instances. To alleviate the difficulty in finding complete agreement, the algorithm uses a fuzziness hyper-parameter, δ. Given that each of four or seven possible answers is treated as a separate class, larger values of δ are used than a binary setting would require. As $\delta < k$, these experiments required a larger pool of observations, and thus were not carried out on the original training-test split put forth by Losada et al. (2020). This algorithm was also compared to MkNNs (Zhang & Zhou, 2007), which does not consider dependencies between output variables. As shown in Table 3.4, the results are comparable to those obtained by the first approach, with a slight increase in ADODL and a small decrease in AHR. As in the previous experiments, the models seem to favor larger values of k. This is consistent with the high variation in answers to several questions discussed earlier. There seems to be little difference in performance between the MkNN and DkNN algorithms both overall and in each model. One possible shortcoming of DkNN in this context is the requirement of exact correspondence in the answers to other questions. However, relaxing this into a proximity requirement might encounter the same smoothing issues arising from aggregating too many neighbors.

An additional limitation of the approach thus far is that the similarity between textual productions is computed on their entirety. By aggregating over the entire textual productions, some localized

Table 3.5 Results of the proposed models on predicting BDI answers in five fold cross validation using DkNN and MkNN.The results are the best for each metric attained by the model, with the corresponding value of (k, δ, D, t) in parentheses.

		MkNN				DkNN			
Model	Token	AHR	ACR	ADODL	DCHR	AHR	ACR	ADODL	DCHR
TF-IDF	trigrams	.377	.698	.821	.344	**.383**	.697	.813	.322
		(20,20,10,10)	(9,9,10,10)	(9,9,10,5)	(7,7,10,5)	(20,10,5,5)	(20,10,5,5)	(5,3,5,5)	(9,7,5,5)
TF-IDF	stems	**.382**	.697	.813	.322	.377	.698	.821	.344
		(20,20,5,5)	(20,20,5,5)	(5,5,5,5)	(9,9,5,5)	(20,10,10,10)	(9,7,10,10)	(9,7,10,5)	(7,5,10,5)
LDA	trigrams	.357	.687	.836	.411	.357	.687	.836	.411
		(5,5,10,20)	(7,7,20,50)	(30,30,20,10)	(7,7,10,5)	(5,3,10,20)	(7,5,20,50)	(30,10,20,10)	(7,5,10,5)
LDA	stems	.363	.702	**.858**	**.567**	.363	.702	**.858**	**.567**
		(20,20,5,5)	(5,5,20,20)	(7,7,20,20)	(7,7,20,20)	(5,3,20,50)	(5,3,20,50)	(7,5,20,20)	(7,5,20,20)
DAN	trigrams	.374	.700	.824	.367	.374	.700	.824	.367
small		(30,30,20,20)	(40,40,10,5)	(7,7,10,5)	(9,9,2,1)	(30,10,20,20)	(40,10,10,5)	(7,5,10,5)	.(9,7,2,1)
DAN	stems	.377	**.705**	.829	.378	.378	**.705**	.829	.378
small		(20,20,20,5)	(40,40,20,5)	(40,40,2,1)	(7,7,20,5)	(40,10,20,5)	(40,10,20,5)	(40,10,2,1)	(7,5,20,5)
DAN	trigrams	.381	.703	.831	.378	.381	.702	.834	.356
large		(20,20,2,1)	(9,9,10,10)	(40,40,20,10)	(30,30,10,5)	(20,10,2,1)	(5,3,10,20)	(20,10,10,5)	(30,10,5,5)
DAN	stems	.366	.697	.836	.378	.366	.697	.836	.333
large		(9,9,5,5)	(9,9,5,5)	(40,40,5,5)	(9,9,20,5)	(9,7,5,5)	(9,7,5,5)	(40,10,5,5)	(5,3,20,5)

similarities might be lost, where a finer-grained approach could find them. Arranging the writings in each textual production in chronological order, we partition them into D parts such that all the partitions in a textual production are of roughly equal word count. We compute a finer-grained similarity by pairing each of the partitions of the textual production considered and averaging over the highest t similarities. Table 3.5 reports the results obtained in cross validation by the DkNN and MkNN algorithms using this similarity, with $(D, t) \in \{(2, 1), (5, 5), (5, 10), (10, 5), (10, 10), (10, 20), (20, 10), (20, 20), (20, 50)\}$. Performance improves overall across all metrics. All encoders appear to favor larger values of D in most metrics. Nonetheless, larger values of D incur greater computational complexity, both in the form of separate calls to the relevant encoders and in terms of comparisons, which grow quadratically with respect to D.

3.3.6 Conclusion

This work studied an approach to predict the potential answers to the BDI of a person from similarity measures between the textual productions of different persons. In particular, the experiments sought to compare the effectiveness of two methods of establishing similarity: using unsupervised extraction of topics and authorship attribution through the use of neural encoders. The proposed system achieves interesting results, indicating that the authorship attribution task can induce a similarity measure useful for depression symptom detection. The task proved difficult nonetheless, as a single similarity measure must account for the prediction of different signs and symptoms of depression. For these reasons, future work will be aimed at adapting the similarity between textual productions to the aspect of depression considered.

Reproducibility. The source code of the proposed systems is licensed under the GNU GPLv3. The datasets are provided on demand by the eRisk organizers.

3.3.7 Acknowledgements

This research was enabled in part by support provided by Calcul Québec and Compute Canada. MJM acknowledges the support of the Natural Sciences and Engineering Research Council of Canada [NSERC Grant number 06487-2017] and the Government of Canada's New Frontiers in Research Fund (NFRF), [NFRFE-2018-00484].

3.4 Postface

The work presented in this chapter sought to arrive at assessments of depression symptoms from text with limited training data using a nearest-neighbours approach. From a pool of subjects with known BDI scores, new BDI scores were inferred based on textual similarity between their writing histories. A measure of similarity based on authorship verification models was proposed and compared against word-count and topic extraction baselines.

Experiments indicated that authorship verification can induce such a measure of similarity. However, approaches yielding comparable results in depression symptom assessment did not necessarily score highly in authorship verification. Small amendments to the overall prediction approach were made in an effort to improve results, namely considering the joint prediction of items and the partitioning of writing histories to allow for local similarities to emerge between segments.

Despite these amendments, an important limitation of the approach is the coarser-grained view that the bag-of-words premise of DANs takes. Indeed, foregoing token order can fail to capture more subtle and complex local cues. While partitioning may mitigate this to some extent, this partitioning is done *a priori* and does not address local patterns within partitions. In an attempt to remediate this, the work presented in the following chapter proposes a model related to DANs that accounts for token order.

CHAPTER 4
POSITION ENCODING SCHEMES FOR LINEAR AGGREGATION OF WORD SEQUENCES

4.1 Preface

As argued in Chapter 3, DANs are a compelling approach to learning representation over large amounts of text given their computational complexity and scalability. However, their bag-of-words premise disregards token order, potentially discarding valuable information to inform these representation. It is then reasonable to ask whether their structure could be amended to account for token order while maintaining the aforementioned advantages. As described in Section 1.3.2, accounting for token order can be done statically, by integrating positional information into token representations, or dynamically, by the order of their integration into the representation of the overall sequence. *A priori*, both approaches permit computational complexity that is linear in the length of the sequence. Nonetheless, the dynamic approach formally requires computation that is sequential along this length and cannot therefore be carried out in parallel. Hence, in practice, RNNs are slower than parallelizable approaches even when their theoretical computational complexity is comparable.

Consequently, directly integrating positional information into token representations may be preferable. Given the linear approach to aggregation on which DANs are built (averaging token representations) careful consideration must be taken in ensuring that positional information is preserved through aggregation and that it can be incorporated meaningfully in the sense that it operates in the manner that is expected. To this end, the present contribution made use of synthetic tasks intended to isolate the effects of position by having them be solvable by accounting for token positions and by that alone.

These experiments were then complemented by experiments in NLP. Such experiments were intended to measure whether these amendments to DANs are indeed helpful in NLP representation, especially given that DANs can obtain creditable results without them.

4.2 References

Diego Maupomé, Fanny Rancourt, Maxime D. Armstrong, Marie-Jean Meurs (2021). Position Encoding Schemes for Linear Aggregation of Word Sequences. *Proceedings of the* 34th *Canadian Conference on Artificial Intelligence. (pp.1-10).* Publisher: Canadian Artificial Intelligence Association. `https://doi.org/10.21428/594757db.37d7654d`

Open-access publication: this article is licensed under a Creative Commons Attribution International license (CC BY 4.0).

Open-source code: the source code of the proposed systems is available here `https://gitlab.labikb.ca/ikb-lab/nlp/canadian-ai-2021/pdan` and is licensed under the GNU GPLv3.

4.3 Publication

Position Encoding Schemes for Linear Aggregation of Word Sequences[1]

Diego Maupomé, Fanny Rancourt, Maxime D. Armstrong, Marie-Jean Meurs

Université du Québec à Montréal, Montréal, QC, Canada

4.3.1 Abstract

DANs show strong performance in several key NLP tasks. However, their chief drawback is not accounting for the position of tokens when encoding sequences. We study how existing position encodings might be integrated into the DAN architecture. In addition, we propose a novel position encoding built specifically for DANs, which allows greater generalization capabilities to unseen lengths of sequences. This is demonstrated on decision tasks on binary sequences. Further, the resulting architecture is compared against unordered aggregation on sentiment analysis both with word- and character-level tokenization, to mixed results.

[1] This article has been modified to match the format of this thesis

4.3.2 Introduction

Modeling natural language requires the decomposition of utterances into tokens, be they words or characters. However, language is sparse. Actually observed utterances make up an extremely small portion of the possible combinations of tokens. It is therefore apparent that in order to make a prediction on an entire phrase, the tokens that comprise it should be aggregated in a manner that is mindful of their role in the phrase. One key aspect of the roles of tokens is their order. Even in highly synthetic languages, the ordering of words can be used to convey information that the words themselves do not carry. As such, much work in NLP has been devoted to accounting for word order in modeling utterances. One such example is the decomposition of phrases into overlapping subsequences of words, called n-grams. These n-grams can then be treated in a structure-agnostic manner because they carry some sense of structure within them. They are however represented symbolically, so their semantic and structural sense is still acquired from surrounding n-grams. Moreover, the number of possible n-grams grows exponentially with respect to the length considered.

An alternative approach to the preservation of token order is to construct said order dynamically. That is, instead of attempting to consume the structure as a whole and then provide a summary of it, one might choose to construct that summary *as* the structure is being consumed in a predetermined order. Such is the case of RNN (Jordan, 1986) approaches, which read a phrase one word at a time and update their numerical *state* (see (Yang et al., 2018; Maupomé & Meurs, 2020)). The final state would then represent a summary of the phrase relevant to the task. Given the non-linear nature of the function updating the state, the order of parsing of the tokens is embedded in this final state.

More recently, Transformer networks (Vaswani et al., 2017) have forgone recurrence altogether. The position of tokens is instead provided by a position encoding, a function mapping a natural number to a real vector, which is added to the *symbol encoding* (e.g. word embedding) of the token. Thereafter, tokens are compared pairwise by the Self-Attention mechanism (Cheng et al., 2016).

Transformers constitute the state of the art for many NLP benchmarks, often in the form of large models pre-trained on unannotated corpora by various schemes (Devlin et al., 2018; Liu et al., 2019; Clark et al., 2020). Nonetheless, such models can be computationally costly to train as well as detrimental to the environment (Schwartz et al., 2019). Crucially, parameter count notwithstanding, the complexity of Transformers is quadratic with respect to the sequence length. While dilated variants aimed at longer documents have been proposed, (Dai et al., 2019), we are interested in reducing the computational cost at the core of the encoder. To this end, we take interest in DANs (Iyyer et al., 2015), which have achieved competitive results in text classification (Cer et al., 2018). They are less computationally intensive than Transformers or RNNs as they operate on the average of the token representations. However, this aggregation does not account for token order, and as such, neither does the encoder. We therefore set out to investigate the following:

1. Whether DANs can be amended to account for word positions in an *efficient* and *generalizing* manner
2. Whether the resulting architecture exhibits benefits over DANs in natural language tasks

To this end, in Section 4.3.3 we study how positional encodings can be integrated into DANs. Section 4.3.4 measures how the resulting models fare in synthetic tasks centered on exploiting positional information. Further, a position encoding more apt at generalizing to unseen lengths of sequences is introduced. In Section 4.3.5, the resulting architecture is evaluated on natural language tasks. Finally, Section 4.3.6 concludes this paper.

4.3.3 Integrating Existing Position Encodings

Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ be a sequence of word vectors, forming a sentence, indexed by $i : 1 \leq i \leq n$. A DAN will compute the sentence representation as follows:

$$\operatorname{dan}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \operatorname{ffn}\left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\right),$$

where ffn denotes a feed-forward network. Thus, DANs average the representation of words making up a sentence and feed this representation through a deep feed-forward network. Our goal is to supplement the word representations to account for their position in the sequence. While Transformer encoders (Vaswani et al., 2017) integrate the position of a word to its symbol embedding by addition, this is not feasible for DANs. Indeed, if we were to simply add the position encoding to the symbol encoding, the linear nature of the average operation on which DANs are predicated would cause the sentence representation to lose track of which positions are assigned to which words. This holds for any position encoding.

We propose instead to integrate the position vector to the symbol vector in one of two ways, by an element-wise product, or a feed-forward network. Let $*$ denote the element-wise multiplication of vectors. The sentence representations computed by the proposed Positional DAN (p-dan) would be given by

$$\text{p-dan}_*(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \text{ffn}\left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i * \boldsymbol{p}(i)\right)$$

in the product case, and by

$$\text{p-dan}_{\text{ffn}}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \text{ffn}\left(\frac{1}{n}\sum_{i=1}^{n} \text{ffn}(\boldsymbol{x}_i + \boldsymbol{p}(i))\right)$$

in the feed-forward network case. While the feed-forward approach is more expressive than the product one, it also has the disadvantage of adding to the parameter count and computational burden of the model. We conduct experiments on synthetic data to evaluate the capacity of these approaches to discern the position of tokens in a sequence.

4.3.4 Experiments on Synthetic Data

We propose three different decision tasks on strings of characters aimed at establishing the capacity to distinguish the positions of characters in a string of different position encodings and integrations in a p-dan. The first task requires two distinguished characters and asks to decide whether the last instance of one of these characters arrives before the last instance of the other. Note that both of these characters can occur multiple times in the string. We refer to this task as the **precedence** task. The second task requires simply to decide whether the string is a palindrome, and we refer to it as the **palindrome** task. The final task is more complex and also requires two distinguished characters. The model is asked to decide whether the last sequence of consecutive occurrences of one of these characters uninterrupted by the other is of count equal to the converse. We refer to this task as the **parity** task. Table 4.1 presents some examples of strings and their associated targets for each task.

To enhance the difficulty of the tasks and increase the proportion of eligible strings, we use an alphabet of only two characters. Indeed, sequences containing several occurrences of a few possible characters will require the model to separate these occurrences by position without the possibility of relying on simple associations between characters and positions. For each task, we generate 50k strings of length 32, randomly selecting 10k for validation, in a stratified manner. The strings

Table 4.1 Examples of binary strings and their associated targets for each of the proposed decision tasks. For the precedence task, in this example, the model must decide whether the last `0` precedes the last `1`.

task	string	target
precedence	1000$\underline{10}$	0
	1100$\underline{01}$	1
	0101$\underline{10}$	0
	010$\underline{0}$1$\underline{1}$	1
palindrome	010010	1
	001001	0
parity	111001	0
	100010	1
	110011	1
	001111	0

generated are not unique. Therefore, we remove any strings present in the training set from the validation set. Strings are generated in equal proportions for both classes. In the case of the parity task, the two possible negative cases (one of the two distinguished symbols occurring more or viceversa) are also generated in roughly equal proportions. Furthermore, in the case of the palindrome task, all non-palindrome strings are the repetition of a substring to prevent the models from relying on the parity of character counts to approximate an answer. Finally, in order to isolate the effects of different position encodings and integrations, the symbol embedding vectors, $\boldsymbol{x}_i$, remain fixed for these experiments. They are random, dense, orthogonal vectors.

We begin with experiments comparing the integration of position encodings with each of two position encodings: sinusoidal position encodings (Vaswani et al., 2017) and learned position embeddings (Gehring et al., 2017). The dimension of the input space is set to 64. Further, all models have three dense layers of 64 units processing the aggregated string. This yields models with 12k parameters for p-dan$_*$ and 25k for p-dan$_{\text{ffn}}$. Each layer uses ReLU activation (Hahnloser & Seung, 2001). The models are trained by the Adam optimizer (Kingma & Ba, 2014) over 30 epochs, and we report the best results obtained in validation. We compare both of the proposed integration approaches to simple additive integration and no positional information, fully expecting these to fail all three tasks. The results are presented in Table 4.2. As expected, only the proposed positional

Table 4.2 Validation accuracy (%) on the three synthetic tasks for the proposed integrations of learned position embedding vectors and sinusoidal position encoding. Random performance is 50% for all tasks.

		Task		
Encoding	Integration	precedence	palindrome	parity
Sinusoidal	$\boldsymbol{x}_i$	57.5	50.2	51.83
	$\boldsymbol{x}_i + \boldsymbol{p}(i)$	57.2	50.3	50.0
	$\text{ffn}(\boldsymbol{x}_i + \boldsymbol{p}(i))$	100.0	98.6	50.0
	$\boldsymbol{x}_i * \boldsymbol{p}(i)$	100.0	100.0	50.0
Embedding	$\boldsymbol{x}_i$	57.6	50.6	52.4
	$\boldsymbol{x}_i + \boldsymbol{p}(i)$	57.4	50.2	50.0
	$\text{ffn}(\boldsymbol{x}_i + \boldsymbol{p}(i))$	100.0	50.1	50.0
	$\boldsymbol{x}_i * \boldsymbol{p}(i)$	98.7	99.8	68.0

integrations succeed at the precedence task. The above-random performance of models without positional information is likely due to their classifying strings with a higher count of `1`s as positive. On the other hand, only the multiplicative approach succeeds at the palindrome task for both position encodings. All models perform poorly at the parity task.

These first results notwithstanding, a more important aspect of these encodings and integrations is whether they successfully generalize to previously unseen lengths. This is unlikely for position embedding vectors, for example. These are "named" vectors with no parameter sharing. Therefore, there is, a priori, no mechanism in place that would allow to generalize to previously unseen lengths. To this end, we introduce a new position encoding in the following section.

4.3.4.1 Normalized Relative Distance

Recently, *relative* position encodings for Transformers (Dai et al., 2019) have increased in popularity. This is in line with the pairwise contextualization effected by the self-attention mechanism that Transformers are predicated on: each word need not be aware of the absolute position of the word it is matched with, only the relative position. However, so far, our approach has been better fit to global positional information, as the aggregation over the sequence happens at once, without any pairwise transformation. Nonetheless, we describe how, by reasoning in terms of relative positions, we can obtain a global position encoding suited for our approach.

Suppose we wished to supplement the word or token representations with those of neighbouring words. Further, we wish for this amended representation to account for the relative position of neighbours. Given a relative position encoding, $\boldsymbol{r} : \mathbb{Z} \longrightarrow \mathbb{R}^d$, the representation of each word, $\boldsymbol{x}_i$, is replaced by a softmax-weighted sum of the sequence:

$$\boldsymbol{x}_i \leftarrow \sum_{j=1}^{n} \frac{e^{\boldsymbol{r}(j-i)}}{\sum_{k=1}^{n} e^{\boldsymbol{r}(k-i)}} * \boldsymbol{x}_j \tag{4.1}$$

Here, exponentiation is applied element-wise. When averaging over an entire sequence $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, we obtain:

$$M = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{e^{\boldsymbol{r}(j-i)}}{\sum_{k=1}^{n} e^{\boldsymbol{r}(k-i)}} * \boldsymbol{x}_j$$

In doing so, the sum can be rearranged in the following manner:

$$M = \frac{1}{n} \sum_{i=1}^{n} \left(\boldsymbol{x}_i * \sum_{j=1}^{n} \frac{e^{\boldsymbol{r}(i-j)}}{\sum_{k=1}^{n} e^{\boldsymbol{r}(k-j)}} \right) \tag{4.2}$$

In other words, the order of summation can be swapped so that each token is weighted by the sum total of the influence the token has on every other token. This is illustrated in Figure 4.1.

We set $\boldsymbol{r}$ as $\boldsymbol{r}(d) = d\boldsymbol{w}$, where $\boldsymbol{w} \in \mathbb{R}^d$ is a vector of learned parameters. Thereby, given that softmax normalization is translation invariant, the position of the *observing* token (*i.e.* the one replaced in Equation 4.1) is irrelevant. That is, it is implied by the positions of neighbors. Thus, Equation 4.2 can be reduced to the encoder we propose, Normalized Relative Distance (NRD) p-dan:

$$\text{p-dan}_{nrd}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n; \boldsymbol{w}) = \sum_{i=1}^{n} \frac{e^{i\boldsymbol{w}}}{\sum_{k=1}^{n} e^{k\boldsymbol{w}}} * \boldsymbol{x}_i$$

Negative components of $\boldsymbol{w}$ will favor the left-hand side of a sequence, and positive ones will favor the right-hand side. Larger absolute values will favor further positions. We initialize $\boldsymbol{w}$ by sampling from a zero-centered truncated normal distribution.

Like that of the other p-dan encoders studied earlier, the complexity of NRD is linear in the length of the sequence. Contrary to sinusoidal position encodings, a shift in position cannot be expressed as a linear map. However, this is consistent with our initial construction of the modified word representations, where a shifted word would need to know what words caused the shift and how

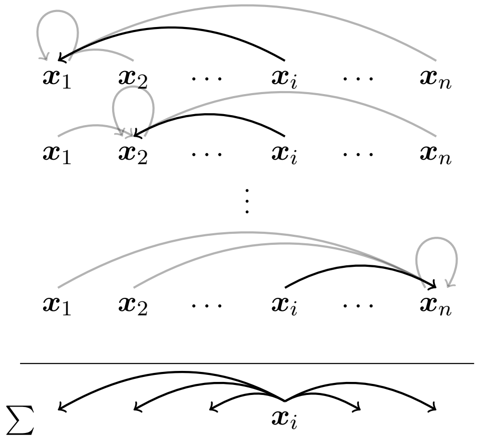
x_1 x_2 $\cdots$ x_i $\cdots$ x_n
x_1 x_2 $\cdots$ x_i $\cdots$ x_n
$\vdots$
x_1 x_2 $\cdots$ x_i $\cdots$ x_n
$\sum$
x_i

Figure 4.1 The proposed position encoding weights each token as a cross-normalized sum of its contributions to other tokens.

close they are. Moreover, this position encoding requires not only to know the position of a word with respect to the start of the sequence but also with respect to its end. This makes intuitive sense, as the role of a word in a sentence is easier to establish if the distance to both beginning and end of said sentence is known.

We proceed with experiments on the previously described synthetic tasks. With the same experimental settings described earlier, this new approach achieves accuracies of 100%, 99.53% and 97.6% on the precedence, palindrome and parity tasks. Further, in order to test the generalization capabilities of this approach and the ones studied earlier, we carry out experiments varying the lengths of the sequences considered. These experiments compare NRD to p-dan$_*$ with sinusoidal position encoding as well as position embeddings. In one set of experiments, the models are trained on sequences of lengths varying from 16 to 32 and validated on sequences of lengths from 33 to 48. These experiments are then repeated with the ranges of sequence lengths reversed, thus assessing the capabilities of the models of transposing the task to longer and shorter sequences. All other settings are identical to those described earlier. The results, presented in Table 4.3, show that for these synthetic tasks, NRD are far more apt at generalizing to unseen lengths of sequences.

However, while these experiments show the capability of the proposed approach to differentiate the positions of tokens and separate subsequences, it is our main goal to test the pertinence of these capabilities on natural language data. The experiments carried out in NLP are presented in the following section.

Table 4.3 Validation accuracy (%) on the three synthetic tasks when generalizing to unseen lengths (longer|shorter). Positions are integrated by element-wise multiplication (p-dan$_*$) for sinusoidal position encoding and position embeddings. Random performance is 50% for all tasks.

	Task					
Encoding	precedence		palindrome		parity	
Sinusoidal	52.1	50.1	49.0	51.0	51.3	51.2
Embedding	50.3	50.8	50.3	50.5	50.4	50.1
NRD	100.0	100.0	82.9	86.4	96.6	95.9

4.3.5 Natural Language Experiments

Following Iyyer et al. (2015), the proposed approach is evaluated on three sentiment analysis datasets. All three datasets consist of film reviews. The Rotten Tomatoes dataset (RT) (Pang & Lee, 2005) contains 11k sentences from film reviews classified as either positive or negative in equal proportions. The larger IMDb dataset (IMDB) (Maas et al., 2011) contains a total of 50k reviews divided evenly between positive and negative ones. Finally, while the Stanford Sentiment Treebank (SST) (Socher et al., 2013), made up of 11k reviews, offers a finer-grained syntax-tree sentiment annotation, we take only the label associated with the whole sentence. For all datasets, given the equal proportions between the negative and positive classes, it is customary to report the binary classification accuracy.

4.3.5.1 Masked Language Model Pretraining

In order to enhance the performance of neural models on natural language data, it is common practice to *pre-train* them on some unsupervised task. We employ **Mask Language Model (MLM)** with dynamic masking (Liu et al., 2019). MLM consists in obscuring tokens in a sequence and asking the model to produce those hidden tokens. As such, it is not a proper language model (a probability distribution over sequences of words) as it is a discriminative model, unlike recurrent language modeling (Mikolov et al., 2010). Nonetheless, MLM, in different forms, has enjoyed much success as a pre-training scheme for Transformers. Given the large size of the output space (the size of the vocabulary) normalizing the loss function for each training instance can be computationally expensive. Instead, one can use *negative sampling* methods, i.e. comparing the true obscured token against only a sample of negative candidates, rather than the entire vocabulary. We employ

Table 4.4 Test set accuracy (%) and parameter counts on the three sentiment analysis tasks for our model and related approaches (where available).

Tokenization level	Model	Size	RT	SST	IMDB
Word	dan	13.3M	76.7	77.1	88.1
	p-dan$_{nrd}$		77.0	76.8	88.1
	pre-trained p-dan$_{nrd}$		77.5	77.2	87.9
Character	dan	937k	50.0	59.1	50.0
	p-dan$_{nrd}$		69.1	70.1	77.3
	pretrained p-dan$_{nrd}$		74.1	73.0	80.0
Word	dan(Iyyer et al., 2015)	—	77.3	83.2	88.8
	dan(Cer et al., 2018)	—	72.2	77.5	—

a variant of noise contrastive estimation amounting to ranking the true token above the noise tokens (Jozefowicz et al., 2016). In the discriminative case, this requires weaker assumptions about the nature of the model than matching the true candidate against each negative one (Ma & Collins, 2018).

We perform MLM on the WikiText-103 dataset (Merity et al., 2016), comprising selected Wikipedia articles totalling over 103 million words. We restrict the original vocabulary of 268k words to the 50k most frequent ones. The training procedure is as follows. For each training step, we create a batch by sampling 16 documents, from each of which 8 sequences of 128 words are sampled. Following (Liu et al., 2019), these sequences are formed by taking a sample of contiguous sentences to fill the set length. The 1024 noise candidates are sampled from the 20k most frequent words (the 100 most frequent excluded) according to their frequency. For efficiency, this sample is taken preemptively and is shared across sequences in a training step. At each step, 16 tokens in each sequence are masked at random. The models are trained by the Adam optimizer (Hahnloser & Seung, 2001), applying early stopping as per the validation loss.

4.3.5.2 Sentiment Analysis

We compare the proposed approach to DANs. All models operate on random word embeddings of size 256. The feed-forward network consists of 7 layers with ELU activation (Clevert et al., 2016).

The models are trained by the Adam optimizer by batches of 64 observations over 20 epochs.

Initial results in classification show little difference between p-dan and DAN models. p-dan models overfit the training set quite heavily, requiring aggressive regularization in the form of dropout to improve performance on the validation sets. This is also the case for pre-trained models. In order to address this issue and exploit positional information, we extend the approach to the character level. That is, each word is treated as a sequence of characters encoded separately by a word encoder. These encoded words are then treated as regular word vectors by a sentence encoder. By descending to the character level, the input space decreases dramatically in size and so do the parameter count of models, decreasing the chance of overfitting. In MLM pre-training, the premise remains the same, only candidate words are sequences of characters encoded by the same word encoder used for the sentence. Our expectations for these experiments were a clearer difference in performance between DAN and p-dan models. Moreover, we expected pretraining to have a greater impact on classification performance as models will learn deeper word representations.

The results for all models and the associated parameter counts are presented in Table 4.4, in addition to results reported in previous work. Our results are consistent with those described by Iyyer et al. (2015). However, our word-level DAN does not achieve the results reported on the SST dataset but is closer to those reported by Cer et al. (2018). This difference might be due to the fact that our experiments do not use word dropout. As previously mentioned, word-level models achieve comparable performance, whether they include positional information or not. This is also consistent with the conclusions drawn by Iyyer et al. (2015), where a DAN achieves comparable performance to a recursive neural network.

As for character-level models, while a drop in performance from DANs was to be expected, interestingly, they perform on par with random labeling on the RT and IMDB datasets. Further, while there is a greater benefit to pre-training for these models, the performance still falls short of that achieved by word-level models. Nonetheless, given the small size of the models, the results obtained are interesting. Given the overfitting we observed from p-dan at the word level, it seems p-dan are much more parameter-efficient than regular DAN.

4.3.6 Conclusion

The work presented has studied the integration of position information in averaging sequence encoders. Such encoders have the advantage of computational complexity that is linear in the length of the sequences. Moreover, a position encoding constructed with such encoders in mind has been introduced. This encoder shows greater generalization capabilities than the other approaches considered on synthetic data specifically conceived to evaluate the discernment of positions. On sentiment analysis, the proposed approaches do not seem to offer any advantages over unordered aggregation, when tokenizing text at the word level. This is consistent with the literature. At the character-level, given the small size of our models, DAN failed to progress beyond random performance, contrary to positional models. Nonetheless, while working at the character level dramatically decreases the parameter count, it of course increases the length of sequences. As such, future work could involve tokenization between the word and character levels, such as byte-pair encodings (Sennrich et al., 2016). Further, a more detailed comparison in terms of results and computational burden with Transformers and the recently proposed Performer (Choromanski et al., 2021), which offers complexity linear with respect to sequence length by approximating full self-attention, is needed. More importantly, future work in NLP could focus on tasks where word order might be more decisive, such as natural language inference or linguistic acceptability.

Reproducibility: The source code of the proposed systems is licensed under the GNU GPLv3.

4.4 Postface

The work presented in this chapter sought to amend the DAN approach to account for the position of tokens. The approaches studied in this contribution provided useful insights into the practicability of the desired approach. The synthetic experiments—those generalizing to sequences of unseen lengths—provided a clear view of the intentional success of the proposed approach: to learn to account for the position of tokens in a sequence. Nonetheless, albeit admittedly limited in scope, natural language experiments yielded mixed results, failing to show any clear gains from the addendum. Other factors could be at play, such as insufficient pretraining data or an inadequate choice of pretraining task or parameter initialization heuristics. Still, the approach was not explored any further.

CHAPTER 5
CONTEXTUALIZER: CONNECTING THE DOTS OF CONTEXT WITH SECOND-ORDER ATTENTION

5.1 Preface

The approach proposed in the previous chapter, positional DANs, are a computationally efficient approach to sequence modelling. However, results in NLP tasks were mitigated. One possible structural reason for these limitations is the compression of information in the token aggregation step. Token vectors are averaged together before being fed through a FFN. While subsequent layers of this network may produce features from information around the entire sequence after the sequence has been aggregated, such computations happen *after* this information has been compressed into a single vector. Moreover, this aggregation is naive in the sense that all tokens are given equal standing by the averaging operation. Thus, tokens are aggregated together without accounting for their role in the sequence. Nevertheless, accounting for this role requires information about the entirety of the sequence. This circular dependency is at the heart of the work presented in this chapter. It proposes a neural encoder akin to DANs that allows for the revisiting of the weight of tokens in aggregation using an attention mechanism.

5.2 Reference

Diego Maupomé, Marie-Jean Meurs (2022). Contextualizer: Connecting the Dots of Context with Second-Order Attention. *Information* 2022, 13(6):290 `https://doi.org/10.3390/info13060290`

Open-access publication: this article is licensed under a Creative Commons Attribution International license (CC BY 4.0).

Open-source code: the source code of the proposed systems is available here `https://gitlab.labikb.ca/ikb-lab/nlp/contextualizer` and is licensed under the GNU GPLv3.

5.3 Publication

Contextualizer: Connecting the Dots of Context with Second-Order Attention[1]

Diego Maupomé, Marie-Jean Meurs

Université du Québec à Montréal, Montréal, QC, Canada

5.3.1 Abstract

Composing the representation of a sentence from the tokens that it comprises is difficult, because such a representation needs to account for how the words present relate to each other. The Transformer architecture does this by iteratively changing token representations with respect to one another. This has the drawback of requiring computation that grows quadratically with respect to the number of tokens. Furthermore, the scalar attention mechanism used by Transformers requires multiple sets of parameters to operate over different features. The present paper proposes a lighter algorithm for sentence representation with complexity linear in sequence length. This algorithm begins with a presumably erroneous value of a context vector and adjusts this value with respect to the tokens at hand. In order to achieve this, representations of words are built combining their symbolic embedding with a positional encoding into single vectors. The algorithm then iteratively weighs and aggregates these vectors using a second-order attention mechanism, which allows different feature pairs to interact with each other separately. Our models report strong results in several well-known text classification tasks

5.3.2 Introduction

The representation of natural language utterances is a central issue of the application of machine learning techniques to natural language. Indeed, natural language occurrences are difficult to represent using the mathematical objects on which algorithms may operate. Manually constructed symbolic representations tend to "leak", to be unable to capture edge cases, leading to the favoring of learned representations (Manning & Schutze, 1999; Ferrone & Zanzotto, 2020). However, it is difficult to learn compact, efficient representations. Language is very sparse: not only does a vast array of local patterns exist, but they will often combine in very few and variable ways. This is not

[1] This article has been modified to match the format of this thesis

an issue of the granularity of fragmentation. For example, one could choose to break up sentences into characters rather than words. In doing so, the vocabulary of base tokens is greatly reduced in number but data grows sparser. More importantly, larger patterns remain difficult to sanction. That is, whether one opts to break sentences up into characters or words, words will still exist, and only a select few combinations thereof will be conceivable, fewer still will be observed.

As such, it is difficult to establish ways in which to construct suitable utterance representations from base components. Some, such as bag-of-words representations will opt to be deliberately simplified and eliminate the need for the learning of this construction. Other, more complex ones, such as topic models, will allow for some learning of utterance representations but require dedicated learning objectives. In contrast, through backpropagation, neural network approaches allow the learning of complex procedures for constructing utterance representations while still allowing for variety in downstream training objectives. Regardless, while the exact parametrization of these operations is to be inferred from the data, there is still considerable structure to provide. That is, the operations in the neural network and their composition need to be specified. Some aspects of this structure are somewhat imposed by practical issues, such as the ability to handle variable length data. Other, more deliberate choices are informed by prior beliefs about language. For example, Tree-structured recursive neural networks are predicated on the notion that complex sentences are recursively constructed from their parts (Socher et al., 2010; Bowman et al., 2015).

As mentioned, language is sparse and combinatorially difficult: even in highly synthetic languages, combinations of words will carry some semantic sense that the words themselves do not carry. This can be broadly construed as an issue of *context*: parts of utterances need to be put into the context of the whole in order to be understood.

Contextualization is fundamentally difficult because it is a circular problem. One cannot recompose a whole by putting its parts in the context of said whole without already knowing what the whole is. One potential solution to this is to iteratively adjust the context against which the parts are compared. That is, to begin with a presumably erroneous value of the context representation and adjust this value with respect to the tokens at hand. This is what Transformer encoders (Vaswani et al., 2017) do: First, the tokens in a sequence are compared against each other by the Self-Attention mechanism (Cheng et al., 2016). This consists in having each token *attend* over all tokens in the sentence, with a parametric attention function producing a scalar weighting of the importance of each token with respect to the attender. Then, a new representation for each token is produced through this weighting. The process is then repeated a set number of times, as shown in Fig. 5.1. In doing so, the word representations produced by this encoder are put into the context of the whole. Transformers have achieved much success in various NLP tasks (Devlin et al., 2018; Cer et al., 2018; Clark et al., 2020; Floridi & Chiriatti, 2020), ranging from sentiment analysis to question-answering and natural language inference (Wang et al., 2019b).

Nonetheless, the Self-Attention mechanism on which Transformers are built has two chief disadvantages. Firstly, because all word pairs are evaluated, the complexity is quadratic with respect to the length of the utterance. Secondly, the weighting provided by the Self-Attention mechanism is based on bilinear forms, mapping each pair of word vectors to a single scalar. As such, Transformers require multiple sets of Self-Attention parameters, called *heads*, so that separate heads might focus on different features of the word vectors. To address these issues, we propose a new architecture - the *Contextualizer* - based on iteratively adjusting a context vector using a second-order attention mechanism. Its computational complexity grows linearly with respect to the sequence length, as opposed to quadratically.

This article is organized as follows. Section 5.3.3 introduces the proposed approach. Section 5.3.5 describes experiments conducted in a few well-known document classification tasks and the results obtained. Finally, Section 5.3.6 concludes this article.

5.3.3 Contextualizer

The proposed encoder utilizes the approach illustrated in Figure 5.1. It proceeds as follows: Over a set number of steps, token representations are matched to the context representation to produce a contextualized representation. These representations are then aggregated into a new context representation, and the process begins anew. The present section details how these computations are carried out in a general-purpose setting.

Let $w_1, \ldots, w_n$ be a sequence of tokens forming a document of length n, indexed by $i = 1, \ldots, n$. Before the contextualization steps, each token is mapped to a single real vector combining informa-

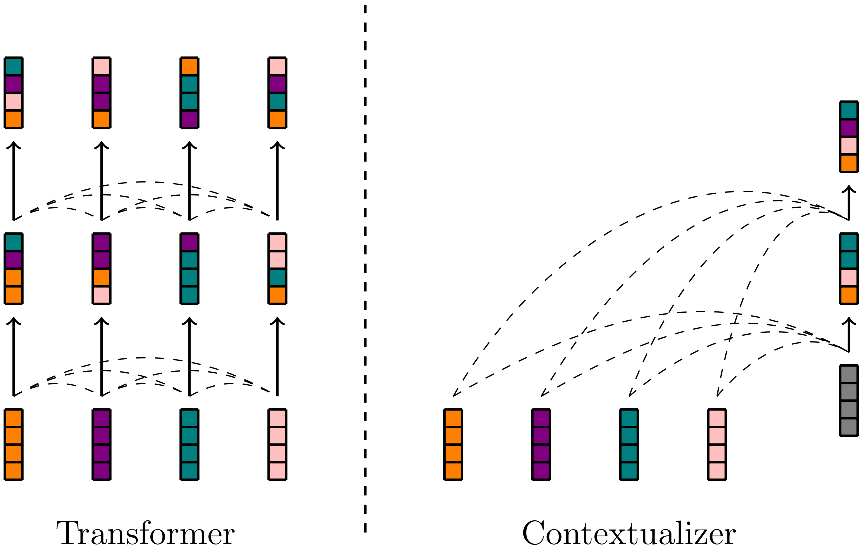

Figure 5.1 The Transformer encoder updates the representations of tokens with respect to each other over a set number of steps by letting each token attend over all tokens in the sequence. This amounts to a distributed representation of the context. The proposed approach updates a single context vector which attends over the tokens at hand.

tion about its identity and position in the sequence. The former is provided by a symbol embedding (*e.g.* pretrained word vectors) of dimension m, $\boldsymbol{e}(w_i) \in \mathbb{R}^m$. The latter is based on a positional encoding inspired by Maupomé et al. (2021b). This positional encoding is as follows: given a vector of parameters, $\boldsymbol{s} \in \mathbb{R}^m$, the j-th component of the encoding for position i, $\boldsymbol{p}(i)$, is given by:

$$\boldsymbol{p}(i)_j = \frac{\exp{(is_j)}}{\sum_{i'=1}^{n} \exp{(i's_j)}}.$$

Multiplicative constants amplify or dampen the peak of a softmax application. By applying the softmax across tokens in the sequence, the parameter vector $\boldsymbol{s}$ allows the model to modulate certain positions for different components of $\boldsymbol{p}$. Combining these two aspects, the token and its position, the vector representation of token w_i is:

$$\boldsymbol{x}_i = \boldsymbol{e}(w_i) * \boldsymbol{p}(i).$$

where $*$ denotes the Hadamard product.

Next, there are K contextualization steps, indexed by $k = 1, \ldots, K$. Each of these steps will produce

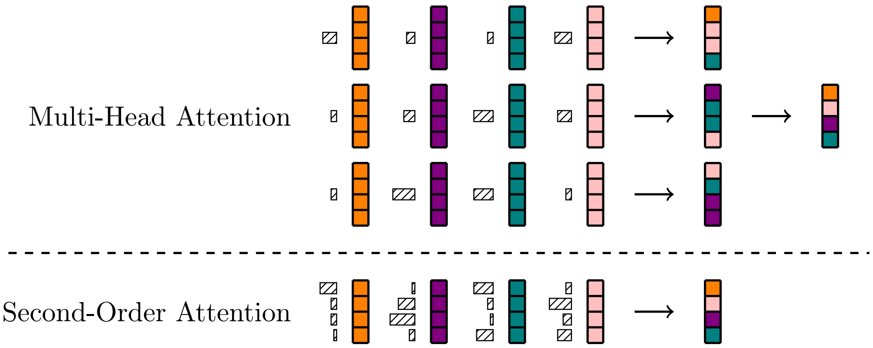

Figure 5.2 Scalar attention requires the use of different sets of attention parameters—heads—to attend to different features of the tokens being aggregated; vector attention allows features to be weighted independently.

a new context vector, $\boldsymbol{c}^k$. The default context used at the first step of contextualization, $\boldsymbol{c}^{(0)}$, can be set to a constant or a learned parameter, for example. This context vector will contextualize the tokens, which will then be aggregated into a new context. An attention mechanism provides the contextualizing function called at every iteration. Using any of the various attention mechanisms in the literature, contextualizing each token would amount to producing a scalar weight, α_i, for each token depending on its content and that of the attender (the context vector in our case). The contextualization of token $\boldsymbol{x}_i$ at step k with respect to the previous context, $\boldsymbol{c}^{(k-1)}$, would then be

$$(\boldsymbol{c}^{(k-1)}, \boldsymbol{x}_i) \mapsto \alpha_i^{(k)} \boldsymbol{x}_i.$$

However, the use of scalar attention weights requires that each component in the operands interacts only with its homolog, collapsing all information to a single number. One must therefore compute several of these interactions with different sets of parameters – called attention heads – so that each of these may focus on different features. This is particularly important when using distributed token representations, where each component might carry a different semantic sense. As such Transformers contain several attention heads. In contrast, to have the weight of each token be a vector, $\boldsymbol{\alpha}_i$, rather than a scalar, α_i, would let each component of the token representation have a separate salience with respect to the current context:

$$(\boldsymbol{c}^{(k-1)}, \boldsymbol{x}_i) \mapsto \boldsymbol{\alpha}_i^{(k)} * \boldsymbol{x}_i, \tag{5.1}$$

This is illustrated in Fig. 5.2.

Such a mechanism would eliminate the need for several heads, as each feature of each token can interact with each feature of the context by a different parameter. However, a second-order attention weighting would require parametrization by a tensor of degree three (3), which would take the parameter count of the model to $O(m^3)$, as both the input and the context vectors are of dimension m. For token representations of even modest size, this would result in a computationally intensive model. Moreover, if the transformation to be learned does not require a full-rank degree-three tensor, such a parametrization would ostensibly be prone to over-fitting because of its excess capacity. Instead, a tensor of rank[2] u can be used, with u becoming a hyper-parameter (see Rabanser et al., 2017; Sutskever et al., 2011; Maupomé & Meurs, 2020). Using this approach, the attention vector for token $\boldsymbol{x}_i$ would be computed as:

$$\boldsymbol{\alpha}_i^{(k)} = \mathbf{W}^{(k)}(\mathbf{U}^{(k)}\boldsymbol{x}_i * \mathbf{V}^{(k)}\boldsymbol{c}^{(k-1)}) + \boldsymbol{b}^{(k)},$$

where $\mathbf{U}^{(k)}, \mathbf{V}^{(k)} \in \mathbb{R}^{u \times m}$ and $\mathbf{W}^{(k)} \in \mathbb{R}^{m \times u}$ are the matrices of parameters for the k-th contextualization step, and $\boldsymbol{b}^{(k)}$ is the corresponding bias vector.

The newly computed attention vectors will serve to update the context vector. This update $\boldsymbol{z}^{(k)}$, is then obtained by adding the contextualized token vectors together, followed by layer normaliza-

[2] Not to be confused with the degree or order of a tensor, the rank of a tensor is analogous to the rank of a matrix.

Table 5.1 Complexities of common layers used in NLP, l designates the kernel size, h, the number of attention heads, n, m and u, designate the length of the sequence, the dimension of the word representations and the multiplicative dimension, respectively. For Transformers and Contextualizers, the complexity is for a single contextualization step.

Layer	**Complexity**
Recurrent	$\mathcal{O}(nm^2)$
Convolutional	$\mathcal{O}(lnm^2)$
Transformer Encoder	$\mathcal{O}(hn^2m)$
Contextualizer	$\mathcal{O}(num)$

tion (Ba et al., 2016):

$$\tilde{\boldsymbol{z}}^{(k)} = \sum_{i=1}^{n} \boldsymbol{\alpha}_i^{(k)} * \boldsymbol{x}_i$$
$$\boldsymbol{z}^{(k)} = \text{LayerNorm}(\tilde{\boldsymbol{z}}^{(k)})$$

The update is then applied to the context vector to obtain the new value for the context vector:

$$\boldsymbol{c}^{(k)} = \boldsymbol{c}^{(k-1)} + \boldsymbol{z}^{(k)}$$

As mentioned, this process is repeated over a set number of steps, allowing information from different sets of tokens to inform the context. The final context vector, $\boldsymbol{c}^{(K)}$, contains then a fixed-size summary of the sequence of tokens at hand.

By reducing sequences of arbitrary length to fixed-size encodings, the proposed approach could potentially squash some information, whereas Transformers encode their input into a sequence of vectors. In return, the number of comparisons in one iteration of the Contextualizer algorithm grows linearly with respect to the number of tokens, as opposed to quadratically for the Transformer. Table 5.1 presents the computational complexities of these two algorithms as well as recurrent and convolutional layers. In addition, as illustrated by Figure 5.1, the Transformer has the drawback of losing sight of the original representation of the tokens, whereas the Contextualizer does not.

5.3.4 Related Work

The computational complexity of Transformers makes them unwieldy for long sequences. As such, there have been several efforts to simplify the computation of full token-to-token Self-Attention to a lighter, more computationally efficient version.

For example, Self-Attention can be limited to local neighborhoods (Parmar et al., 2018). That is, instead of comparing tokens attend to each other throughout the sequence, tokens can be restrained to attending over a local portion of the sequence. This approach can be complemented by having sliding-window neighborhoods (Beltagy et al., 2020; Chelba et al., 2020). It can also be combined with *causal masking*: allowing tokens to attend only over preceding tokens. In doing so, segments can be chained recursively, allowing deeper contextualization levels to receive information from earlier segments (Dai et al., 2019).

More sophisticated, dynamic approaches can rely on inferred neighborhood. Rather than determine neighbors by position, tokens can be bucketed by locality-sensitive hashing (Kitaev et al., 2020) or clustering (Roy et al., 2020). Alternatively, neighborhood can be determined by the syntax tree of the utterance at hand (Bai et al., 2021).

Yet another approach is to replace softmax Self-Attention with a lighter variant. For example, by replacing the exponential kernel implicit to softmax self-attention by a polynomial kernel, key-value products can be shared across queries, reducing computational complexity (Katharopoulos et al., 2020). In the same vein, softmax Self-Attention can be approximated via random feature maps, thus reducing the dimension of the attention space as a function of sequence length rather than eliminating token pairs (Choromanski et al., 2021).

5.3.5 Experiments and Results

5.3.5.1 Exploratory experiments

We begin with experiments on the well-known Rotten Tomatoes dataset (MR) (Pang & Lee, 2005)[3]. It consists of 11k English-language sentences from film reviews classed as either positive or negative

[3] Released in June 2005, Available at `https://www.cs.cornell.edu/home/llee/papers/pang-lee-stars.home.html`

Table 5.2 Test accuracy (%) on the MR task and computation time (ms/batch) for Transformer and Contextualizer models of similar sizes. The parameter counts exclude word-piece embeddings. Batch size of 32.

	Contextualizer		**Transformer**	
Parameter count	Accuracy	Time	Accuracy	Time
0.5M	73.5	57	72.4	118
1.0M	74.3	96	74.9	164
1.5M	73.4	120	73.0	223
2.0M	74.0	151	74.7	265

in equal proportions. The documents are fairly short, with 95% of them being 45 words long or shorter.

The first experiments seek to compare the test set classification accuracy and computation time of Transformer and Contextualizer models of comparable sizes. These parameter counts are chosen to be relatively small in accordance with the limited size of the dataset. For each architecture, four models of 0.5, 1, 1.5 and 2 million parameters are trained and tested. These counts exclude the initial token embedding layer. Following Transformer approaches (Devlin et al., 2018; Liu et al., 2019; Floridi & Chiriatti, 2020), documents are tokenized into word-piece tokens (Schuster & Nakajima, 2012; Merity et al., 2016). The hyperparameters of the models are set following Vaswani et al. (2017) while adjusting for the smaller model size. The dimension of the embedding space, m, is set to 128. The number of contextualization steps (the number of encoding layers in the Transformer) is set to 5 for all models. The number of attention heads for Transformers is set to 4. Hence, the variable adjusted to increase the parameter count of the models is the dimension of the attention space for Transformers and the rank of the tensor decomposition, u, for Contextualizers.

Models were trained on a Intel Core i7-7700 machine with 32GiB of memory over 10 epochs with batches of 32 examples using the Adam (Kingma & Ba, 2014) optimizer, with a learning rate of 1E-4. The best model on a 10% validation set over the 10 epochs is selected for testing.

Results are presented in Table 5.2. As shown, both architectures show comparable results across model sizes. As expected, computation time is much greater for Transformer networks.

Table 5.3 Test accuracy (%) on the MR task for different default context strategies

	K	
$\boldsymbol{c}^{(0)}$	1	5
$\mathbf{1}$	73.1	71.2
$\boldsymbol{c}_d$	73.5	72.2
$\mathcal{U}(-\mathbf{1},\mathbf{1})$	57.9	72.4

We then proceed with experiments measuring the effect on performance of the nature of the default context. All Contextualizer models share the same configurations except the default context, which is set to be either a constant, $\boldsymbol{c}^{(0)} = \mathbf{1}$, a vector of learned parameters, $\boldsymbol{c}^{(0)} = \boldsymbol{c}_d$, or a random vector redrawn for every document from a uniform distribution, $\boldsymbol{c}^{(0)} \sim \mathcal{U}(-\mathbf{1},\mathbf{1})$. We hypothesize that using a random default context will make the network more robust by reducing dependence on prior beliefs and therefore mitigating overfitting. For the same reasons, one could expect a learned default context to be more likely to overfit than a constant one.

Table 5.3 summarizes the results. As one might expect, a random starting context vector hurts performance when contextualization is performed but once. The models are quick to adjust, as all choices of default context seem to arrive at very similar final accuracies.

5.3.5.2 Further results

We continue with experiments in binary document classification on other well-known English-language datasets in order to compare the performance of the proposed approach to the Transformer-based Universal Sentence Encoder architectures (USE) (Cer et al., 2018). The Subjectivity dataset[4] (SUBJ) (Pang & Lee, 2004) comprises 10k sentences around films classed as subjective or objective, released in June 2004. Annotation is automatic based on whether the sentence is a synopsis (objective) or an appreciation (subjective). The Customer Reviews dataset (CR)[5], introduced by Hu & Liu (2004), comprises 3775 reviews of electronic products. These reviews were extracted from Amazon and CNET and manually annotated. They are equally divided between positive and

[4] Available at `https://www.cs.cornell.edu/people/pabo/movie-review-data/`

[5] Available at `http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar`

Table 5.4 Test accuracy (%) on several benchmark text classification tasks of our Contextualizer models compared to the Universal Sentence Encoder architectures (USE), Transformer-based (T) and Deep Averaging Network-based (D)

Model	**MR**	**CR**	**SUBJ**	**MPQA**
Contextualizer	76.6	79.0	91.2	85.3
USE (T)	81.4	87.4	93.9	87.0
USE (D)	74.5	81.0	92.7	85.4

negative sentiment. Finally, the Multi-Perspective Question Answering dataset[6] (MPQA) (Wiebe et al., 2005) deals again in sentiment polarity, containing 10,606k phrases from press articles.

Test accuracy on several benchmark text classification tasks of our Contextualizer models compared to the Universal Sentence Encoders (USE) (Cer et al., 2018), Transformer-based (T) and Deep Averaging Network-based (D) is shown in Table 5.4. The results of these experiments demonstrate that the Contextualizer architecture can perform competitively, even with small models and relatively small datasets.

5.3.6 Conclusion

We have proposed an algorithm for constructing sentence representations based on the notion of iteratively adjusting a central context vector. This algorithm is closely related to the encoder part of the Transformer algorithm. One key difference is the use of the proposed second-order attention mechanism, replacing multiple attention heads.

Another important difference is the computational complexity, which is linear in sequence length. Transformer models have been the driving force behind the expansive use and development of large models such as BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), GPT-3 (Floridi & Chiriatti, 2020), Electra (Clark et al., 2020), among others, which are extensively trained by adapted language-modeling tasks. The reduced complexity of the Contextualizer model would be of use both in terms of pre-training and in terms of wielding these large models in downstream tasks. This is important given how the computational cost of these large models can further the economical divide

[6] Available at `https://mpqa.cs.pitt.edu/corpora/mpqa_corpus/`

between low- and high-resource laboratories and companies (Luitse & Denkena, 2021). Additionally, this computational demand also incurs significant environmental impact (Strubell et al., 2019). Therefore, lighter-computation approaches could help mitigate these concerns.

Yet, as seen in Section 5.3.5, the Contextualizer achieves results comparable to those of a Transformer when controlling for model size. Our approach can also achieve competitive results in benchmark document classification tasks even with low parameter counts. Furthermore, our results suggest the approach is robust to different choices of the number of contextualization steps and default contexts. Further work will be conducted in this direction as well as in formally characterizing the conditions that stabilize the context vector as the number of contextualization steps increases.

5.4 Postface

The work presented in this chapter sought to propose a neural text encoder. Like the Transformer, it proceeds by iteratively adjusting the encoding over a set number of steps. The proposed encoder, the Contextualizer, like DANs has computational complexity linear with respect to sequence length. This resulted in lower computation time with comparable model sizes in our experiments. Nonetheless, a clear advantage of Transformer encoders is the built-in scaling of sequence representation. Indeed, both Contextualizer and DAN encoders will transform sequences of any length into a fixed-size representation, potentially forcing the compression of information. In contrast, Transformers will encode input sequences into sequences of equal size. In so doing, the encoding function inferred is not required to account for compression of information of variable quantities.

CHAPTER 6
EARLY DETECTION OF SIGNS OF PATHOLOGICAL GAMBLING, SELF-HARM AND DEPRESSION THROUGH TOPIC EXTRACTION AND NEURAL NETWORKS

6.1 Preface

This chapter presents an account of our participation to a mental health and NLP shared task (Parapar et al., 2021). This task presents a series of different challenges asking participants to make automated assessments from social media text around specific aspects of mental health. Such tasks offer particular constraints not otherwise found in research, thus granting opportunities for creative solutions to these challenges. In particular, this shared task included BDI prediction, allowing the deployment of the approaches studied in Chapter 3.

6.2 References

Diego Maupomé, Maxime D. Armstrong, Fanny Rancourt, Thomas Soulas, Marie-Jean Meurs (2021). Early Detection of Signs of Pathological Gambling, Self-Harm and Depression through Topic Extraction and Neural Networks. *CLEF 2021 Conference and Labs of the Evaluation Forum. pp. 1031–1045* `https://ceur-ws.org/Vol-2936/paper-83.pdf`

Open-access publication: this article is licensed under a Creative Commons Attribution International license (CC BY 4.0).

Open-source code: the source code of the proposed systems is available here `https://gitlab.labikb.ca/ikb-lab/nlp/erisk2021` and is licensed under the GNU GPLv3.

6.3 Publication

Early Detection of Signs of Pathological Gambling, Self-Harm and Depression through Topic Extraction and Neural Networks[1]

[1] This article has been modified to match the format of this thesis

Diego Maupomé, Maxime D. Armstrong, Fanny Rancourt, Thomas Soulas, Marie-Jean Meurs

Université du Québec à Montréal, Montréal, QC, Canada

6.3.1 Abstract

The eRisk track at CLEF 2021 comprised tasks on the detection of problem gambling and self-harm, and the assessment of the symptoms of depression. RELAI participated in these tasks through the use of topic extraction algorithms and neural networks. These approaches achieved strong results in the ranking-based evaluation of the pathological gambling and self-harm tasks as well as in the depression symptomatology task.

6.3.2 Introduction

This paper describes the participation of the RELAI team in the eRisk 2021 shared tasks. Since 2017, the eRisk shared tasks have aimed to invite innovation in Natural Language Processing and other artificial intelligence-based methods towards the assessment of possible risks to mental health based on online behavior (Parapar et al., 2021). In 2021, three tasks were put forth. The first task (Task 1) introduced the problem of detecting the signs of pathological gambling. The second task (Task 2), a continuation of Tasks 2 and 1 from 2019 and 2020, respectively, focuses on the assessment of the risk of self-harm. Finally, continuing from Tasks 3 and 2 from 2019 and 2020, Task 3 asked participants to predict the severity of depression symptoms as defined by a standard clinical questionnaire.

6.3.3 Task 1: Early Detection of Signs of Pathological Gambling

Pathological gambling is a public health issue with prevalence rate between 0.2% and 2.1% (Abbott, 2020). Accessing treatment is difficult since general practitioners usually do not screen for this pathology (Achab et al., 2014) and, by the time the issue has become evident, the patient has lost control (Haefeli et al., 2011). With the rise of online platforms, more data are available for potential detection systems (Braverman et al., 2013). Recently there has been research work focused on communications with customer services to detect whether a subject was at risk of gambling (Haefeli et al., 2015, 2011). However, to the best of our knowledge, textual productions from online fora have not yet been used to detect early signs of pathological gambling.

6.3.3.1 Task and Data

As mentioned, the data issue from Reddit users. These subjects have been labeled as either at risk for pathological gambling (positive) or not (negative). No labeled data were provided for training models. As such, the following pertains solely to the test data.

The test data comprised 2348 subjects, 164 of which were positive (6.9%). The test data are released iteratively, with each step counting at most one writing per subject. These writings are sorted in chronological order of publication. The first iteration includes writings from all test subjects. Thereafter, subjects are included as long as they have unseen writings.

Algorithms are expected to predict both a binary label and a score at each step. The label can default to negative. However, a positive prediction for a given user is binding, and all label predictions thereafter are disregarded. Evaluation, which is detailed in the following subsection, considers the labels and the timeliness of positive predictions, as well as the scores.

6.3.3.2 Evaluation

Performance is measured both on the ultimate decision made on each subject, using binary classification metrics, and the predicted scores, using ranking metrics. The classification metrics include standard precision, recall and the associated F_1 score, as well as **Early Risk Detection Error (*ERDE*)**, $latency_{TP}$, *speed* and $F_{latency}$. In addition to accounting for the binary prediction on a given subject, $ERDE$ seeks to account for the timeliness of that prediction by counting the number of writings processed by the predicting algorithm before producing a positive prediction. Given t_u, p_u, respectively the ground truth and predicted labels for a given subject, k_u the number of processed writings, and a given threshold o, the $ERDE$ is computed as:

$$ERDE_o(p_u, t_u, k) = \begin{cases} c_{fp} & \text{if } p_u \neq t_u = 0 \\ 1 & \text{if } p_u \neq t_u = 1 \\ \frac{1}{1+e^{o-k}} & \text{if } p_u = t_u = 1 \\ 0 & \text{otherwise} \end{cases}$$

Here, c_{fp} is a constant set to the rate of positive subjects in the test set. This per-subject $ERDE$ is averaged across the test set, and is to be minimized. Thus, false negatives are counted as errors, and false positives are counted as a fraction of an error in proportion to the number of positive subjects. The delay in decision is only considered for true positive prediction, where a standard sigmoid function counts the number of writings processed, k, offset by the chosen threshold, o. As with the other classification metrics used, true negatives are disregarded. $ERDE$ was evaluated at $o = 5$ and $o = 50$.

Likewise, $latency_{TP}$ measures the median delay in true positive predictions:

$$latency_{TP} = median\{k_u : u \in U, p_u = t_u = 1\}$$

Similarly, *speed* and $F_{latency}$ (Sadeque et al., 2018) are computed by penalizing this delay albeit in a smoother manner:

$$speed = 1 - \text{median}\,\{penalty(k_u) : u \in U, p_u = t_u = 1\}$$

$$F_{latency} = F_1 \cdot speed.$$

The individual penalty is given by a logistic function and depends on a scaling parameter, p:

$$penalty(k_u) = -1 + \frac{2}{1 + e^{-pk_u - 1}}$$

The scores attached to each subject are used to rank them. This ranking is evaluated by standard information retrieval metrics: P@10, $NDCG$@10 and $NDCG$@100. These are evaluated after 1, 100, 500 and 1000 writings have been processed.

6.3.3.3 Approaches and Training

Having no annotated training data at hand, data available on the web were exploited for both training and prediction. Two authorship attribution approaches were put forward for this task. In both cases, our approaches assess whether a test user belongs to a set of gambling testimonials using a similarity distance measure between their textual productions. A test user u is said to be at risk of pathological gambling if the minimal similarity distance δ^u_{min} computed for them is smaller than a threshold θ.

Since topic modeling has shown good potential in such authorship attribution task (Maupomé et al., 2021a), it is selected to represent both test users' textual production and the testimonials. Given the performances of the **Embedding Topic Model (ETM)** (Dieng et al., 2020) in (Armstrong et al., 2021), this model is selected for topic extraction. Our ETM model is trained on a corpus made from two datasets. The first part is made from the textual productions from the Subreddits *Problem Gambling*[2] and *Gambling Addiction Support*[3], ensuring the presence of gambling-related vocabulary in the corpus. The second part is made up of control subjects from the 2018 eRisk depression dataset, adding general topics to the corpus. Both gambling-related and control content were added in equal part to this novel training corpus in order to limit any discrepancies.

The ETM is trained following the methodology described in (Armstrong et al., 2021). Using the trained model, the test user's textual productions and the testimonials are mapped to a vector of topic probabilities, which are then used in our two authorship attribution approaches to compute the similarity. Here, the similarity is given by computing the Hellinger distance between two vectors of topic probabilities.

6.3.3.4 Testimonials

Our first approach consists in using testimonials found on the Web to assess the pathological gambling risk of the users from the test data. The testimonials offered by 199 compulsive gamblers were found on *Gambler's Help*[4]. These testimonials are considered to be our testimonials set

[2] `https://www.reddit.com/r/problemgambling/`

[3] `https://www.reddit.com/r/GamblingAddiction/`

[4] `https://gamblershelp.com.au/`

$T = \{t_1, \ldots, t_{199}\}$.

Here, we aim to find the minimal similarity distance threshold θ_{min} to be considered at risk of pathological gambling by using the testimonials set T. Then, every testimonial $t \in T$ is compared to the others using a one-against-all cross validation technique, *i.e.* 1 vs. 198 testimonials, to compute its distance δ^t from every other testimonial. A testimonial t is represented by its vector of topic probabilities $\boldsymbol{ect}$, which allows computing the Hellinger distance between testimonials t_i and t_j, as

$$\delta^{t_i,t_j} = Hellinger(\boldsymbol{ect}_i, \boldsymbol{ect}_j)$$

By doing so, it is possible to find the maximal similarity distance obtained for a testimonial compared to all the others, as

$$\delta^{t_i}_{max} = max(\{\delta^{t_i,t_j}, \ldots, \delta^{t_i,t_{n-1}}\})$$

Assuming that every testimonial has to be part of the testimonial set, the maximal similarity distance obtained across the evaluation is then the minimal threshold to be part of the set, as

$$\delta_{max} = max(\{\delta^{t_1}_{max}, \ldots, \delta^{t_n}_{max}\}) \qquad \theta_{min} = \delta_{max}$$

Thus, predicting if a test user pertains to the testimonial set is given by computing the similarity distance of its vector of topic probabilities against the vector of every testimonial. For a given test user, if the minimal similarity distance computed is lower than the threshold, then it is decided that the test user is part of the testimonial set.

$$\delta^{u}_{min} = min(\{\delta^{u,t_1}, \ldots, \delta^{u,t_n}\})$$

$$prediction(\delta^{u}_{min}, \theta_{min}) = \begin{cases} 1 & \text{if } \delta^{u}_{min} \leq \theta_{min} \\ 0 & otherwise \end{cases}$$

One potential issue with the use of these testimonials is that their language might differ from that used in Reddit fora. Nonetheless, topic models should smooth over the particulars by grouping word co-occurrences.

6.3.3.4.1 Questionnaire

Our second approach makes use of a self-evaluation questionnaire in addition to the set of testimonials. The self-evaluation questionnaire, which is often offered by resources for compulsive gamblers,

Table 6.1 Results obtained on the test set of Task 1 by our models and the best performing models on each metric. Runs are denoted APPROACH-stem and APPROACH-reg following the methodology adopted from (Armstrong et al., 2021)

.

System	Run	$precision$	$recall$	F_1	$ERDE_5$	$ERDE_{50}$	$latency_{TP}$	$speed$	$F_{latency}$
Questionnaire-stem	0	0.138	0.988	0.243	**0.048**	0.036	1	**1**	0.243
Questionnaire-reg	1	0.108	**1**	0.194	0.057	0.045	1	**1**	0.194
Testimonials-stem	2	0.071	**1**	0.132	0.067	0.064	1	**1**	0.132
Testimonials-reg	3	0.071	**1**	0.132	0.066	0.064	1	**1**	0.132
CeDRI	1	.070	**1**	.131	.066	.065	1	**1**	.131
UNSL	2	**.586**	.939	**.721**	.073	**.020**	11	.961	**.693**

was found on several websites, including *Gamblers Anonymous Montreal*[5]. This one is composed of 20 questions answerable by yes or no. An individual scoring 7 or more positive answers from this questionnaire is considered at risk of a pathological gambling problem.

Comparably to the testimonial approach, we aim to find the minimal similarity distance threshold θ_{min} to be considered at risk of pathological gambling. Here, this threshold is computed using the self-evaluation questionnaire and the testimonial set T. Given the questionnaire q and its vector of topic probabilities $\boldsymbol{ecq}$, a testimonial t is said close enough to the questionnaire to be considered at risk of pathological gambling if the Hellinger distance between $\boldsymbol{ect}$ and $\boldsymbol{ecq}$ is less or equal to the threshold θ_{min}. Thus, the idea is to find the maximal similarity distance δ^q_{max} to define this threshold, as

$$\delta^q_{max} = max(\{\delta^{q,t_1}, \dots, \delta^{q,t_n}\}) \qquad \theta_{min} = \delta^q_{max}$$

Then, predicting if a test user is at risk of pathological gambling can be made using its distance from the self-evaluation questionnaire, such as

$$prediction(\delta^{u,q}, \theta_{min}) = \begin{cases} 1 & \text{if } \delta^{u,q} \leq \theta_{min} \\ 0 & otherwise \end{cases}$$

Table 6.2 Ranking-based results (P@10; $NDCG$@10; $NDCG$@10) on the test set of Task 1 for our models and the best model

Team	Run	1 writing			100 writings			500 writings			1000 writings		
RELAI	0	.9	.92	.73	**1**	**1**	.93	**1**	**1**	.92	**1**	**1**	.91
	1	**1**	**1**	.72	**1**	**1**	.91	**1**	**1**	.91	**1**	**1**	.91
	2	.8	.81	.49	.5	.43	.32	.5	.55	.42	.5	.55	.41
	3	.8	.88	.61	.6	.68	.49	.7	.77	.55	.8	.85	.55
UNSL	2	**1**	**1**	**.85**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

6.3.3.5 Results

The results are presented in Tables 6.1 and 6.2. Our best model was Run 0, outperforming our other approaches on both precision and F-measure. While showing a limited precision, it obtained the best $ERDE_5$ across every other system presented for this task.

6.3.4 Task 2: Early Detection of Signs of Self-Harm

The task was introduced in 2019, and teams did not have access to any training data, producing modest results (Losada et al., 2019). The following year, Transformer-based approaches were the most prolific, achieving the best precision, F_1-score, $ERDE$s and latency-weighted F_1. XLM-RoBERTa models were trained on texts from the Pushshift Reddit Dataset (Baumgartner et al., 2020), and predicted whether a user was at risk of self-harm or not by averaging on all their known posts. Each of their runs targeted a specific evaluation metric for the fine-tuning. As noted by (Losada et al., 2020), most runs had a near perfect recall and also a very low precision. Of those, NLP-UNED (runs 3 & 4) (Ageitos et al., 2020) seemed to gather the best overall performances. Their systems used a combination of textual features and sentiment analysis from the entire user's historic to predict whether they were at risk. The best results are presented in Table 6.3.

6.3.4.1 Task and Data

The task objective and evaluation process are identical to that of Task 1, including the iterative evaluation of models and the metrics. The key difference, however, is that a training set was

[5] http://gamontreal.ca/

Table 6.3 Summary of the best results obtained on eRisk 2020 T1 (Self-harm).

System	Run	*precision*	*recall*	F_1	$ERDE_5$	$ERDE_{50}$	$latency_{TP}$	*speed*	$F_{latency}$
iLab (Martínez-Castaño et al., 2020)	0	0.833	0.577	0.682	0.252	0.111	10	0.965	**0.658**
	1	**0.913**	0.404	0.560	0.248	0.149	10	0.965	0.540
	2	0.544	0.654	0.594	**0.134**	0.118	2	0.996	0.592
	3	0.564	0.885	0.689	0.287	**0.071**	45	0.830	0.572
	4	0.828	0.692	**0.754**	0.255	0.255	100	0.632	0.476
NLP-UNED (Ageitos et al., 2020)	3, 4	0.246	**1**	0.395	0.213	0.185	**1**	**1**	0.395

provided. The training set counted 145 positive subjects out of 763 (19.0%), while the test set counted 152 positive out of 1448 (10.5%).

6.3.4.2 Approaches

For this task, two approaches based on neural networks were tested. One is based on the Contextualizer encoder (Maupomé & Meurs, 2022), while the other is based on RoBERTa embeddings (Liu et al., 2019).

6.3.4.2.1 Contextualizer

Following (Maupomé et al., 2020), two modes aggregating the different writings in a subject's history were used. The first, nested aggregation, uses one Contextualizer encoder to encode the writings separately into a single vector representation and another Contextualizer encoder to aggregate writings together. The second mode, flat aggregation, performs both of these steps at once by providing positional information to each word about its writing and within-writing position. For both of these approaches, the positional information about writings is not the chronological order but the time difference with respect to the most recent writing.

6.3.4.2.2 RoBERTa embeddings

A Transformer model was trained using RoBERTa (Liu et al., 2019). This training was carried out on Reddit data by masked language modeling. This approach tokenizes writings into character n-grams based on their frequency in the source corpus. Once the Transformer was trained, the writings in the training set were transformed into token embeddings. These token embeddings constituting a writing, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, are averaged together into a single document vector:

$$\bar{\boldsymbol{x}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i$$

In order to combine these document representations into a single vector per subject, we posit that writings farther in the past should be given less importance than more recent ones. Given a set of m documents, $\{(\bar{\boldsymbol{x}}_j, t_j)\}_{j=1}^{m}$, where $t_j \in \mathbb{R}$ denotes the difference in hours from the j-th document to the most recent one, the document vectors are aggregated into a single vector:

$$\boldsymbol{u} = \sum_{j=1}^{m} \boldsymbol{\alpha}^{t_j} * \bar{\boldsymbol{x}}_j$$

Here, $\boldsymbol{\alpha}$ is a vector of learned parameters and the exponentiation and multiplication, $*$, are applied element-wise. This allows for each feature to decay at an independent rate. Thereafter, the predicted probability of having observed a positive instance is given by:

$$\hat{y} = \sigma(\boldsymbol{w}^{\top} \boldsymbol{u}),$$

where σ denotes the standard sigmoid function and $\boldsymbol{w}$ is a vector of learned parameters.

6.3.4.2.3 Training

All models are trained by gradient descent with a binary cross entropy minimization objective using the Adam algorithm (Kingma & Ba, 2014). To compensate for possible discrepancies between the proportions of labels between the training and test sets, a balanced validation set was built taking half of the positive subjects and a number of negative subjects to match. In addition, a stratified validation set was also tested. In training, subjects were inversely weighted in the loss function to account for the imbalance. For both approaches, different contiguous samples of writings from each subject are taken at each epoch. The size of such samples was chosen to allow models to make early decisions without requiring a long history of writings. In validation, however, the most recent documents for each subject are taken. Model selection was based on the area under the precision-recall curve, which is equivalent to the average precision. The selected models are presented in Table 6.4. Except for Run 5, all of the chosen models were validated on the balanced validation set.

Table 6.4 Models selected for the test stage of Task 2. The number of writings indicates how many of the most recents writings per subject the model will consider.

Run	Model	Nb of writings
0	Flat Contextualizer	5
1	Nested Contextualizer	5
2	Roberta Embeddings	20
3	Roberta Embeddings	5
4	Roberta Embeddings (strat. validation)	5

Table 6.5 Classification results on the test set of Task 2 for our models and the best models per metric

Team	Run	*precision*	*recall*	F_1	$ERDE_5$	$ERDE_{50}$	$latency_{TP}$	*speed*	$F_{latency}$
RELAI	0	.138	.967	.242	.140	.073	5	.984	.238
	1	.114	.993	.205	.146	.086	5	.984	.202
	2	.488	.276	.353	.087	.082	2	.996	.352
	3	.207	.875	.335	.079	.056	2	.996	.334
	4	.119	.868	.209	.120	.089	2	.996	.206
Birmingham	0	**.757**	.349	.477	.085	.07	4	.988	.472
CeDRI	2	.116	**1.0**	.19	.096	.094	**1**	**1.0**	.19
UNSL	0	.336	.914	.491	.125	**.034**	11	.961	.472
	4	.532	.763	**.627**	.064	.038	3	.992	**.622**
UPV-Symanto	1	.276	.638	.385	**.059**	.056	**1**	**1.0**	.385

Table 6.6 Ranking-based results (P@10; $NDCG$@10; $NDCG$@10) on the test set of Task 2 for our models and the best models per metric

Team	Run	1 writing			100 writings			500 writings			1000 writings		
RELAI	0	.1	.06	.11	.4	.37	.46	.4	.32	.38	.5	.47	.41
	1	0	0	.12	.2	.12	.36	0	0	.27	.1	.06	.28
	2	.8	.71	.4	.4	.28	.4	**1**	**1**	.6	**1**	**1**	.57
	3	.7	.76	.43	0	0	.31	.9	.88	.59	.8	.75	.56
	4	.4	.44	.34	0	0	.21	.4	.34	.27	.5	.5	.31
UNSL	0	**1**	**1**	**.7**	.7	.74	**.82**	.8	.81	**.8**	.8	.81	**.8**
	4	**1**	**1**	.63	**.9**	.81	.76	.9	.81	.71	.8	.73	.69
UPV-Symanto	0	.8	.83	.53	**.9**	**.94**	.67	.9	.94	.67	0	0	0

6.3.4.3 Results

Classification and ranking-based results are presented in Tables 6.5 and 6.6, respectively. Label decisions were fairly quick and favored positive decisions, resulting in low $latency_{TP}$ (2 to 5). Run 2 notwithstanding, recall was high (>.85), resulting in low precision (<.25) and modest F_1 (<.34) for those runs. This is partly due to the smaller proportion of positive subjects in the test set. Run 2 achieved much higher precision than our other runs (.488) but at the price of low recall (.276) resulting in a comparable F_1 (.353). However, Run 2 seemed to outperform our other runs in ranking-based evaluation and achieving perfect P@10 and $NDCG$@10 with 500 and 1000 writings processed. Its $NDCG$@100 was also high, indicating an adjustment to the decision policy might benefit classification. Overall, as per the ranking-based metrics, the scores produced by our models seemed to improve from 100 to 1000 writings, with the exception of Run 1, which remained low throughout.

6.3.5 Task 3: Measuring the severity of the signs of depression

As described by (Losada et al., 2020), the task consists in mapping a subject's writings to a well-known tool for the assessment of depression symptoms, the BDI (Beck et al., 1996). In 2019, the two approaches that gathered the best performances leveraged the dependency between the severity of depression categories and the severity of the signs. The first aimed to predict the severity category and then deduce the severity of each sign of depression (Burdisso et al., 2019), achieving the most precise predicted answers to the questionnaire. The second system leveraged textual similarity between the user's productions and the questions from the questionnaire to fill it. By combining those answers, the best results regarding the prediction of depression severity were obtained (Abed-Esfahani et al., 2019). The results are presented in Table 6.7. A description of each evaluation metric is provided at Section 6.3.5.2.

For the second iteration of this task in eRisk 2020, the best performances remained similar to those observed in the previous year. The approaches achieving the best results were based on psycholinguistic features (Oliveira, 2020), pre-trained Transformers (Martínez-Castaño et al., 2020), LDA-based authorship attribution (Maupomé et al., 2020), or combining a support vector machine with a radial basis kernel (Uban & Rosso, 2020). The 2020 best results are presented in Table 6.8.

Table 6.7 Summary of the best results obtained on eRisk 2019 T3 (severity)

Run	AHR	ACR	ADODL	DCHR
CAMH_GPT_nearest_unsupervised (Abed-Esfahani et al., 2019)	23.81%	57.06%	**81.03%**	**45.00%**
UNSL (Burdisso et al., 2019)	**41.43%**	69.13%	78.02%	40.0%
	40.71%	**71.27%**	80.48%	35.00%

Table 6.8 Summary of the best results obtained on eRisk 2020 T2 (severity)

Run	AHR	ACR	ADODL	DCHR
BioInfo@UAVR (Oliveira, 2020)	**38.30%**	69.21%	76.01%	30.00%
iLab run2 (Martínez-Castaño et al., 2020)	37.07%	**69.41%**	81.70%	27.14%
prhlt_svm_features (Uban & Rosso, 2020)	34.56%	67.44%	80.63%	**35.71%**
relai_lda_user (Maupomé et al., 2020)	36.39%	68.32%	**83.15%**	34.29%

6.3.5.1 Task and Data

As with Tasks 1 and 2 the dataset comprises a history of writings per subject. However, instead of a binary label, each subject is associated with a set of 21 labels corresponding to the answers they gave to each item of the BDI. Furthermore, evaluation did not include a temporal aspect: the entire history of writings for the test subjects was made available at once. As shown in Fig. 6.1 the BDI scores are overall higher in the test set, with the median and median absolute deviation for the training and test set being (20.0, 9.5) and (27.0, 10.0) respectively.

6.3.5.2 Evaluation

In order to evaluate BDI predictions against the true BDI answers associated with a set of subjects, (Losada et al., 2020) propose four metrics. The Average Hit Rate (AHR) is the rate of exactly correct predictions averaged across the 21 items of the BDI and across subjects. In contrast, the Average Closeness Rate (ACR) measures the proximity in value ([0,3]) between the predicted and true answer when compared to the maximum possible difference (3). Similarly, the Average Difference in Overall Depression Levels (ADODL) compares the total score of the predicted BDI to the true total score, once again normalized by the maximum (63). Finally, the Depression Category Hit Rate (DCHR)

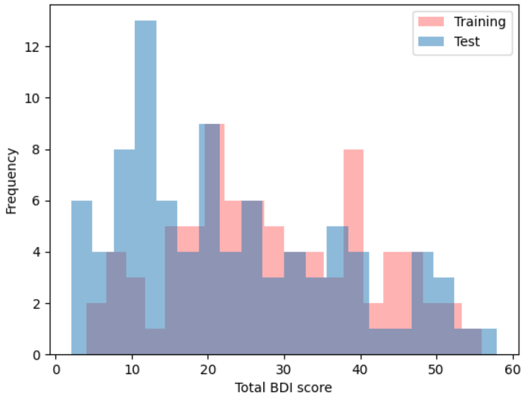

Figure 6.1 Histogram of total BDI scores in the training and test sets

is the accuracy in the depression categorization resulting from the predicted BDIs of subjects.

6.3.5.3 Approaches

Following (Maupomé et al., 2021a), we opt for predicting the BDI items based on the similarities of the writings of the test subjects to those of the training subjects. These similarities are computed on learned representations of the textual production of subjects. These representations were based on topic modeling or neural encoders trained on authorship decision. In addition to the categorical prediction of BDI items proposed by (Maupomé et al., 2021a), a regression approach (Reg.) based on the values of the answers was also tested, with the values being multiplied by the relevant similarity score. There is a high variance in answers even among subjects in the same depression category. To address this in the regression approach, each training subject's answer to each question is smoothed to the average answer in their depression category by a hyperparameter, $\beta \in [0, 1]$,

$$a_j \leftarrow \beta * a_j + (1 - \beta) * \bar{a}_j$$

Here, a_j denotes the answer selected by the j-th training subject to a given item in the BDI and $\bar{a}_j$, the average answer selected by the subjects in the depression category that said subject belongs too.

Variance aside, this approach is still potentially highly sensitive to the particular distribution of BDI scores in the training set. To address this, a nearest-neighbors approach was tested wherein a set number of neighbors was to be drawn from each of the four depression categories. This approach is denoted k'NN, and can be applied in both the regression and categorical settings.

6.3.5.3.1 Topic Modeling

Topic modeling consists in inferring probability distributions over a vocabulary of words, such that the documents, the subjects' histories in our case, constitute a mixture of such distributions. As a baseline, we selected the well-known LDA algorithm. Further, another topic model, operating on word embeddings rather than symbolic word representations like LDA, ETM (Dieng et al., 2020), was also tested. Models were trained on a depression-detection dataset also issuing from Reddit (Losada et al., 2018). This training was carried out considering the entire history of writings from each subject as a single document. For the LDA approach, two tokenization schemes were tested: character trigrams and word stems. In contrast, only stemming was tested for the ETM model for interpretability purposes.

6.3.5.3.2 Authorship decision

DANs were trained to discern whether two sets of writings were authored by the same person. This can induce a representation relevant to depression symptoms (Maupomé et al., 2021a). As with topics models, these models were trained on the eRisk 2018 Depression dataset, using alternately character trigrams and word stems. The training procedure consists in sampling non-overlapping sets of writings from subjects and pairing them together. Pairs of samples issuing from the same subject constitute positive examples and pairs issuing from different subjects, negative ones.

6.3.5.3.3 Model selection

As previously mentioned, this approach is potentially highly sensitive to the distribution of BDI scores in the training set. In order to mitigate this, the validation set selected contained 24 subjects equally divided among the four depression categories defined by the BDI. The hyper-parameter values tested were borrowed from (Maupomé et al., 2021a).

Models were selected based on the performance on all four metrics. Indeed, selecting the top performers for each metric separately might exclude models performing well overall. However, in order to combine all four metrics into a single quantity by which to select models requires

Table 6.9 Models selected for prediction on the test set of Task 3

Run	Encoder	Algorithm	Tokenization	k	δ	D	t
0	DAN	DMkNN	trigram	30	10	1	1
1	DAN	DMkNN	trigram	9	7	10	1
2	ETM	Reg. k'NN	stemming	5	-	1	1
3	DAN	k'NN	trigram	5	-	1	1
4	LDA	Reg. k'NN	trigram	3	-	10	10

Table 6.10 Results on Task 3

Team	Run	AHR	ACR	ADODL	DCHR
RELAI	0	34.64	67.58	78.59	23.75
RELAI	1	30.18	65.26	78.91	25.00
RELAI	2	**38.78**	72.56	80.27	35.71
RELAI	3	34.82	66.07	72.38	11.25
RELAI	4	28.33	63.19	68.00	10.00
DUTH ATHENA	5	**35.36**	67.18	73.97	15.00
UPB	5	34.17	**73.17**	82.42	32.50
CYUT	2	32.62	69.46	**83.59**	**41.25**

consideration. Although the metrics are valued in the unit interval, they have different scales in practice. Therefore, combining the performance for each metric for all models and hyper-parameter values, the z-score for each one was computed. Then, the average z-score across all metrics was used to select the models. The selected models are shown in Table 6.9. As in (Maupomé et al., 2021a), k denotes the number of neighbors considered, while δ is the consensus parameter of DMkNN. D and t denote the size of the writing history partition and the number of parcel pairs considered.

6.3.5.4 Results

The results are shown in Table 6.10. Our best model was Run 2, outperforming the rest of our models on each metric. Overall, the total BDI scores predicted were low, with Run 0 having the highest median of 18 and Run 3 having the lowest of 8. Predictions were quite tight within each run, with Run 1 having the highest median absolute deviation of 6. Furthermore predictions were consistent among runs, with Run 1 and 4 agreeing the least, on only 44% of answers globally. Interestingly, although Run 0 agreed the most with Run 2 (64%), it achieved much weaker results, especially in terms of DCHR.

Overall, the approach remains sensitive to the particulars of the training set where neighbors are sourced. This is perhaps due to text alone not eliciting, by unsupervised learning alone, similarity that pertains to depression symptoms. Future work could include integrating manual annotation or prior knowledge in the training of the similarity models, authorship and topic alike. Moreover, in order for the overall approach to be effective in the 21-way prediction at hand, similarity could be handled separately by component or groups of components of the subject representation.

6.3.6 Conclusion

RELAI participated in all three eRisk 2021 shared tasks. Task 1, *Early Detection of the Signs of Pathological Gambling*, proved an interesting challenge in the lack of training data. Nonetheless, the use of testimonials and self-assessment questionnaires constitutes a promising avenue in such a context. The *Early Detection of the Signs of Self-Harm*, Task 2, was a more conventional one. The favoring of early decisions resulted in high recall but poor precision overall. Nonetheless, some of the proposed approaches produced good ranking-based results. Finally, Task 3, *Measuring the Severity of the Signs of Depression*, remains thoroughly difficult. However, in predicting BDI scores based on similarity, restricting the number of neighbors per depression category proved an interesting option to address uncertain distributions of BDI scores.

The source code of the proposed systems is licensed under the GNU GPLv3. The datasets are provided on demand by the eRisk organizers.

6.3.7 Acknowledgments

This research was enabled by support provided by Calcul Québec and Compute Canada. MJM acknowledges the support of the Natural Sciences and Engineering Research Council of Canada [NSERC Grant number 06487-2017] and the Government of Canada's New Frontiers in Research Fund (NFRF), [NFRFE-2018-00484].

CHAPTER 7
MENTALHEALTHBERT: A PRETRAINED LANGUAGE MODEL FOR MENTAL HEALTH RISK DETECTION

7.1 Preface

Chapter 1 raised multitask learning as a means of improving the quality of data fitting, particularly in terms of representation. This can take the form of semi-supervised learning, wherein functions are first trained on an unsupervised task before serving as a basis for the supervised task of interest. Data for such unsupervised tasks is easier to gather in large quantities. This allows for the training of large capacity models to produce powerful functions. While the computational cost of this training is significant, it is offset by the versatility of its end product, which can then be used in a variety of settings.

Nonetheless, while such models are trained on vast data, they benefit from some form of specialization towards the specific domain on which they will be deployed. This is termed domain adaptation. For example, a function produced from general-purpose customer review data will benefit in its performance from being adapted to the particulars of electronics customer reviews before being deployed in this area. It is then important to ask what this entails for language modelling and mental health. While general-purpose functions will benefit from adaptation to mental health data (Ji et al., 2022), it is not clear how specific these data should be to a given aspect of mental health.

The following contribution delves into this consideration, by training models on data surrounding several mental health concerns and evaluating the performance of ensuing classifiers.

7.2 References

Diego Maupomé, Fanny Rancourt, Raouf Belbahar, Marie-Jean Meurs (2023). A Pretrained Language Model for Mental Health Risk Detection. *Proceedings of Machine Learning for Cognitive and Mental Health Workshop. AAAI 2024. (pp.23–28)* `https://ceur-ws.org/Vol-3649/Paper17.pdf`

7.3 Publication

MentalHealthBERT
A Pretrained Language Model for Mental Health Risk Detection[1]

Diego Maupomé, Fanny Rancourt, Raouf Belbahar, Marie-Jean Meurs

Université du Québec à Montréal, Montréal, QC, Canada

7.3.1 Abstract

Early detection of mental health issues is a key contributor to efficient treatment. Natural language processing-based approaches can provide automated means to facilitate access to appropriate services and support for at-risk individuals. Using pretrained language models provides state-of-the-art results in various downstream tasks as these models leverage significant amounts of textual content. They can be critical in data-scarce research areas, such as early detection of mental health issues. Nonetheless, exposing models to domain-specific language can be beneficial to their performance in downstream task. To this end, we release pretrained language models, MentalHealthBERT, leveraging content from Reddit fora discussing anorexia, depression and self-harm. These models are evaluated on risk detection tasks for the respective conditions.

7.3.2 Introduction

Early intervention in mental health and well-being has become a critical principle of mental health care, ushering in an international wave of service reform (Schotanus-Dijkstra et al., 2017; McGorry & Mei, 2018). Given the ever-growing use and diversity of online social media, there has been a vast increase in research interest for the use of NLP for the development of automated means of analyzing online textual content in the service of mental health care support and early intervention in particular (Shing et al., 2020; Maupomé et al., 2021a).

The inference of such predictive models requires the gathering of annotated data. These data map online textual content to an assessment of certain aspects of the mental health of the authors of

[1] This article has been modified to match the format of this thesis

this content. Such assessments are difficult to produce. Whereas for other common tasks in NLP, annotation can operate on the observation itself (*e.g.* the text), annotation relating to mental health generally requires further information about the author of the textual content. That is, the true aspects of interest pertain to the author rather than the text. In particular, clinically grounded assessments require access to the individual. As such, gathering annotated data is expensive and time-consuming.

In the absence of large quantities of annotated data, it is a well-established principle of machine learning that pretraining on an unsupervised task can help performance on a downstream supervised task. As such, there has been increased interest in the production of pretrained models leveraging large amounts of textual content (Peters et al., 2018; Liu et al., 2019; Clark et al., 2020; Ji et al., 2022). Such models are made available for use on a variety of specialized downstream tasks (Wang et al., 2019b,a). The core tenet is that large models trained on sufficiently large data sets will learn to produce useful representations of text regardless of what specialized task these representations will serve. Such a framework leverages large quantities of data for models to learn aspects of language that are thought to precede the specifics of the specialized task.

While this assumption may hold for *tasks*, pretraining data can also issue from different *sources* than the specialized data. As such, representations produced by general-purpose models might be inadequate. Recent work has pointed to the benefits of domain specificity in large pretrained models. Broadly, the term *domain* refers to the topics, mode or register of documents. Domain specificity concerns can take the form of models pretrained entirely on domain-specific data or domain adaptation.In either case, gains in downstream task performance have been reported for several tasks and domains from the use of such domain-specific pretraining (Gururangan et al., 2020).

The textual data analyzed for mental health care purposes issues from Internet fora and social media. These data can differ both in register and topics from news or encyclopedia articles comprising significant parts of large corpora. Nonetheless, there is no established linguistic consensus on what constitutes a domain (see Plank, 2011, Sec. 3.4.1). Given this difficulty in defining the notion of domain, it is difficult to delineate given domains or to establish quantitative differences between them.

Pragmatically, one might ask whether a more narrow concept of a given domain may provide more benefit to downstream task performance than a broader one. The present work seeks to study this issue in the context of mental health risk assessment. Models are pretrained on data from Internet fora revolving around three different mental health concerns: anorexia, depression and self-harm. The models are evaluated on detection tasks surrounding these concerns and compared to models trained on broader data (Ji et al., 2022). Our results corroborate the benefits of domain adaptation for general-purpose language models but only show advantages to pretraining-data specificity in one case: anorexia.

7.3.3 Data

7.3.3.1 Retrieval

The data were extracted from three Reddit[2] fora (known as *subreddits*): `depression`, `selfharm` and `AnorexiaNervosa`. This extraction was performed using Pushshift[3] (Baumgartner et al., 2020). For all three subreddits, it was limited to posts published from the 1st of January 2019 to the 25th of November 2020.[4] Further, posts struck off as "removed" were discarded. The fields associated with each post include the title and body of the post, as well as the timestamp, the score (aggregation of up- and down-votes), the number of replies and the identifier of the parent post. No additional filtering was applied.

7.3.3.2 Description

All subreddits considered are described as communities that offer a safe place and peer support for people affected by the aforementioned issues[5]. Summary statistics for the corpus are presented in Table 7.1.

The `depression` forum is by far the biggest community of the three with more than 736,000 members as of March 2nd 2021. Of those, about 45% authored at least one publication (*i.e.* a post or a

[2] `https://www.reddit.com`

[3] `https://pushshift.io/`

[4] The latest post from `AnorexiaNervosa` was published on the 3rd of December 2020.

[5] As per their respective "About Community" section of each subreddit

comment) in the selected time frame. A similar proportion can be observed from AnorexiaNervosa. In turn, it jumps to almost two-thirds for selfharm. Across all three subreddits, approximately 40% of the authors published exactly once. Despite having the fewest overall publications, threads on AnorexiaNervosa seem to generate the most engagement, having a higher ratio of comments per thread and remaining active for longer periods. The smaller size of this community is a likely explanation for these observations.

	AnorexiaNervosa	depression	selfharm
Tokens	3.7G	160.9G	18.8G
Vocabulary	38.4k	303.2k	87.1k
Posts	10.3k	412.4k	78.0k
average number of tokens	141	204	116
Comments	45.8k	1404.3k	236.4k
average number of tokens	49	54	41
Unique author	10.1k	338.1k	43.3k
Community size*	23.8k	736k	66.4k

Table 7.1 Subreddits statistics. Unique authors exclude deleted accounts. *As of March 2nd 2021.

7.3.4 Ethical Considerations

All posts collected in the aforementioned subreddits are public but our collection will not be publicly available. Further, resources discussed in this work will be released upon the signature of a User Agreement. The released model should only be used in combination with other screening tools for prevention purposes under the supervision of trained mental health professionals. Hence, this system does not aim to diagnose mental health disorders and should not be used to do so.

However, the misuse of this kind of work can have negative societal impacts. For example, an organization could use our pretrained language models to detect at-risk job applicants of mental health disorders before hiring. This practice, violating the terms of the release agreement, would further spur discrimination in hiring processes in addition to well-documented gender and racial unfairness (Sánchez-Monedero et al., 2020; Quillian et al., 2017).

While this line of research could potentially advance early intervention and treatment processes, it does not directly address the stigma surrounding mental health issues and underlying the high rate of treatment avoidance and discontinuation (Wahl, 2012; Henderson et al., 2013). Further, widespread study and deployment of models in this direction could potentially lead to self-censorship, defeating its purpose.

It is also important to note that demographic data on the authors is missing. As noted by Shatz (2017), most subreddits do not have data regarding their community demographics. Hence, it is impossible to ensure that the textual productions used to train the released model adequately represent content from diverse individuals. To the best of our knowledge, there is no readily available dataset containing information regarding the author's age, gender, ethnicity, or location. From inferred demographics, Amir et al. (2019) presented that those sensitive attributes affect depression prevalence across social media users. Aguirre et al. (2021) observed performance gaps related to gender and racial attributes. To address this gap, a data collection combining strict privacy policies and clinical supervision must be achieved. As noted by Aguirre et al. (2021), storing such sensitive data comes with serious potential harms. Therefore, it is critical to enforce protective measures such as data anonymization.

7.3.5 Pretraining

7.3.5.1 Preprocessing

One key issue in modeling corpora from Internet fora rather than an edited outlet, such as a newspaper or encyclopedia, is the longer vocabulary tail caused by misspellings, neologisms and even usernames. Common practice would be to remove words having fewer than three occurrences (Merity et al., 2016). Keeping such words would increase the computational burden of the model while having little chance of learning because of the limited number of occurrences. However, this is not suitable for our purposes: Important words might be misspelled or obfuscated, but their exclusion will hinder performance (Plank, 2016). Similarly, usernames and neologisms might be composed from familiar, significant words. As such, we preserve the entire vocabulary of each dataset, relying on subword-level tokenization to capture these variations.

Before learning this tokenization, the data was split into training and validation sets by stratifying across length (word count) percentiles. This preserves the key length statistics, such as the median and interquartile range. In terms of vocabulary, words in the validation set not present in the training set make up 0.50%, 0.20% and 0.10% of occurrences in the anorexia, self-harm and depression sets, respectively.

The data was tokenized by Byte-Pair Encodings (BPEs) (Sennrich et al., 2016) at the byte level (Liu et al., 2019), with the merges extracted from all three datasets. This consolidation was done to provide a more robust tokenization scheme, less skewed towards any particular forum, while still learning the words and spellings of online parlance. For comparison, each dataset was also tokenized using merges learned exclusively from itself.

7.3.5.2 Training

Once tokenized, these datasets were used to train Transformers (Vaswani et al., 2017) using the RoBERTa approach (Liu et al., 2019). Models are trained by the Adam optimizer (Kingma & Ba, 2014) with a learning rate of 5E-4 on batches of 256 sequences of a maximum length of 256 tokens. Training takes place over a maximum of 300 epochs, applying early stopping based on validation set perplexity.

7.3.6 Mental Health Risk Detection

We evaluate the MentalHealthBERT models on the eRisk datasets (Losada et al., 2018, 2019, 2020). These datasets comprise Reddit users (subjects) labeled as being at risk (positive) or not (negative) for depression, self-harm or anorexia, respectively. For each subject, a history of their writings is included, spanning a variety of subreddits. The proportion of positive subjects is fairly small and varies somewhat, as does the size of the datasets, as shown in Table 7.2.

The key issue is utilizing the document-level encoding afforded by MentalHealthBERT in predictions at the history level, which spans a variety of presumably independent writings. In order to make this subject-level prediction, information gathered across a set of writings needs to be aggregated. To achieve this, token embeddings are averaged together within posts and subsequently fed to a feed-forward network with a single hidden layer and hyperbolic tangent activation. The resulting document vectors are then aggregated by averaging into a single vector encoding a history of writings.

This vector is then mapped to a binary prediction for the sequence of writings by a feed-forward network with a single hidden layer with hyperbolic tangent activation.

	Train		Test	
dataset	positive	negative	positive	negative
Depression	214	1493	98	1302
Self-Harm	145	618	152	1296
Anorexia	61	411	73	742

Table 7.2 Positive and negative subject counts from the eRisk training and testing sets

7.3.6.1 Experiments

The experiments compare the performance of MentalHealthBERT to the generic RoBERTa Transformer as well as the latter further pretrained on our data (domain adaptation). For MentalHealthBERT, experiments were carried out using BPEs learned from the combined dataset as well as from the individual collections. Additionally, we run experiments using MentalRoBERTa (Ji et al., 2022). This model was pretrained on Reddit data from several different fora touching on mental health topics[6]. It should be noted that the results reported by the authors on the eRisk depression detection task are not comparable to those reported here, as they make use of a custom data split with some resampling (Ji, 2022). Models are evaluated per the area under the precision-recall curve.

One difficulty of detecting potential threats to mental health is the small proportion of positive subjects that can be found in datasets and, indeed, in a real-world setting. Additionally, for the selected datasets, these proportions vary widely between the training and testing sets, as shown in Table 7.2. Models were evaluated using the latest set of data for each task: 2022 for Depression, 2021 for Self-Harm and 2019 for Anorexia. Training and validation sets for each task were obtained by combining the data from all previous sets and randomly selecting 80% of subjects for training and 20% for validation, preserving equal proportions of positive and negative subjects.

To address this class imbalance in training, a number of strategies were deployed, including inverse

[6] An exhaustive list of the fora from which pretraining data were extracted is not available, but they include `depression`, `SuicideWatch`, `Anxiety`, `offmychest`, `bipolar`, `mentalillness`, and `mentalhealth`.

class weighting, class weighting based on effective samples (Cui et al., 2019) and Focal Loss (Lin et al., 2018). These proved to be ineffective in validation. The most effective mechanism proved to be sampling batches of even proportions of positive and negative subjects in training.

The number of writings used to arrive at a prediction for a subject was set to $m = 50$. In order to reduce overfitting, a contiguous sample of m writings was taken per subject at training time. In validation and testing, only the last m writings were taken. The classifiers are trained by the Adam optimizer (Kingma & Ba, 2014) over 10 epochs. Given the relatively modest size of the datasets in terms of positive subjects, only the top two layers of the Transformer encoder were trained, with a learning rate of 1E-5. The remainder of the model had a learning rate set to 1E-4.

Results on the eRisk datasets are presented in Table 7.3. Results for the base RoBERTa model indicate improvements with domain adaptation, in agreement with the literature (Gururangan et al., 2020). Perhaps counterintuitively, these improvements appear to decrease with the amount of domain adaptation data available. MentalRoBERTa and MentalHealthBERT achieve comparable results in all but the anorexia task, for which MentalRoBERTa and domain-adapted RoBERTa outperform MentalRoBERTa. This may be due to a deficiency in eating disorder content in pretraining MentalRoBERTa, though we cannot confirm this. Tokenization seems to be inconsequential, with a more marked decrease in performance for the combined tokenizer in depression. Given the difficulty that specialized tokenization puts in transferring learning, it is difficult to support in light of these results. Finally, there appears to be little difference in performance between domain-adapted RoBERTa and the best MentalHealthBERT model, suggesting no real benefit to training blank models over adapting pretrained models. In light of these results, it is difficult to establish whether the specific domain pretraining of MentalHealthBERT helps downstream performance more so than more the general domain adaptation found in MentalRoBERTa. As mentioned, benefits are only observable for the anorexia task. Given what is known of the pretraining of MentalRoBERTa, it is difficult to establish whether this may be due to any material characteristics of discourse around anorexia or its relatively smaller weight in pretraining.

	Tokenization	Depression	Self-Harm	Anorexia
RoBERTa (Liu et al., 2019)	RoBERTa	0.487	0.434	0.401
RoBERTa with domain adaptation	RoBERTa	0.496	**0.494**	0.555
MentalRoBERTa (Ji et al., 2022)	RoBERTa	**0.536**	0.476	0.416
MentalHealthBERT	Separate	0.520	0.475	0.560
MentalHealthBERT	Combined	0.457	0.485	**0.569**

Table 7.3 Area under the precision-recall curve on the eRisk test sets

7.3.7 Conclusion

There is increased research interest in the development of NLP approaches to assist in early risk assessment in mental health care. Gathering annotated data is a costly process, making pretraining a crucial step in the modeling process. Thus, pretrained language models can be a valuable resource. However, general-purpose language models, while trained on large amounts of data, may not be suited to specific domains, such as mental health discussions. As such, there is interest in adapting language models to particular domains.

In the case of mental health risk assessment from text, domain-specific pretraining resources would contain discourse concerning mental health concerns. However, it is worth considering whether discourse issuing from outlets specific to a particular mental health concern are more adequate than discourse around mental health issues at large. Our experiments have thus made use of data extracted from fora dedicated to specific mental health concerns to pretrain models. These models are compared to general-purpose language models as well as language models pretrained on broader mental health content in a mental health risk assessment task. Our results indicate that domain adaptation does improve classification performance. However, a difference in performance between more narrowly pretrained models is only manifest in anorexia risk detection.

Further work is needed to understand how textual data from separate mental health topics interact in terms of benefits from pretraining: more experimentation is needed to find whether the detection of certain mental health concerns is improved by pooling pretraining data and whether these gains in detection performance align with the comorbidity of the underlying disorders. Were this the case, those benefits might be explained by the mention of related concerns in discussions about a specific mental health concern. While pretraining data for our experiments was extracted from dedicated fora, our experiments do not control for the mention of related disorders or threats to mental health.

7.3.7.1 Release of Resources

Given the sensitive nature of the resources introduced, the models and associated open-source code will be released upon signing of a User Agreement providing details on permitted uses.

CONCLUSION

Given the widespread use of online social media and the substantial data it generates, there has been much research interest in leveraging this data to provide insight into the mental status of individuals. Such analyses could be greatly beneficial in bridging the gap between mental health care and at-risk persons in the form of screening or treatment support tools. A significant portion of this data is in natural language. It has been the object of this thesis to address what we believe to be the key NLP considerations of such an enterprise.

Throughout this thesis, important challenges to the successful utilization of NLP in analyzing the mental status of individuals have been expounded. An immediate challenge is data availability. For machine learning algorithms to learn to map writing histories to an assessment of the mental health of their author, they must be provided with examples. These example analyses must be carried out by clinicians with access to the individuals in question, making the gathering of data costly. Data collections are therefore few in the literature. As an alternative, a number of strategies have been deployed in the literature to circumvent this issue, using more readily available information as a proxy for the clinical signals of interest. The use of these strategies does allow for easier gathering of larger-scale datasets. As such, they have been widely deployed in the literature. Nonetheless, it is unclear how well these proxy signals capture the clinical signals they intend to, thereby compromising the validity of predictors they engender.

This uncertainty bleeds into the design of NLP methods for mental health analysis. Chapter 2 highlighted a particular epistemic consideration: in contrast with many habitual problem settings in NLP, the predictions that are expected concern the individuals who authored the text rather than the text itself. This makes addressing the problem difficult, as the signals of interest may be dispersed through vast amounts of text. The research contributions that have been presented in this thesis have sought to address these challenges. Chapter 3 presented an approach to estimating depression symptoms with low training resources. This approach was based on inducing similarity through authorship decision. While the underlying models were chosen for their ability to scale over large amounts of text, a major concern is that their bag-of-words premise discarded important information. Consequently, Chapters 4 and 5 investigated the development of neural network approaches to representing natural language with computational complexity that is linear in the number of tokens, achieving mitigated results. Building on these ideas, Chapter 6 tests some of these neural network approaches in mental health risk detection. Results indicate the predictors these approaches produce are susceptible to shifts in the distribution between classes. Finally, Chapter 7 presents a contribution investigating language model pre-training. Specifically, it studied whether pre-training on data issuing from fora dedicated to a specific mental health concern improves prediction quality when compared to pre-training on general purpose data.

Mental health assessment tasks proved to be difficult, particularly binary risk detection tasks. These tasks were approached by neural networks encoding a subset of writings per subject. These subsets were kept fairly small in number due to both computing resource considerations and temporal considerations. Indeed, training on small sets of writings was intended to stimulate predictors to make decisions given little information. These sets were sampled randomly at each training iteration as a means of regularization. Nevertheless, performance does seem to deteriorate significantly in testing. The explaining factors are unclear. Resulting predictors may be poorly calibrated, making them vulnerable to shifts in the population balance between classes. Another issue may lie in predictor selection, particularly in early stopping. It is unclear how the sets of writings for validation examples should be selected, or indeed, if they should be larger in size than those used in training. Future work could explore more dynamic policies based on the the particular findings of the predictor. Additionally, another possible limit of the approaches presented is that pre-training was predicated only on writings. A possible future work direction would be to extend this idea into pretraining writing history representations in an unsupervised manner. Such data is readily available and requires minimal annotation. Nonetheless, it is not given that such approaches will be sound in that they will elicit representations that capture interesting aspects of the mental status of persons.

We contend these uncertainties to be the product of a larger issue at play: a limited understanding of the relationship between what NLP methods may capture and the clinical aspects of interest. This is due to NLP methods being deployed as "oracles", which would parse text data and provide an assessment of mental status in habitual clinical terms. This would allow mental health practitioners to be provided supplemental information or confirmation that lies within established clinical theory. In turn, NLP practitioners are content to take on this challenge if it means calquing the problem on well known task archetypes. This convenience makes this (somewhat tacit) premise inviting.

Nonetheless, it makes approaches difficult to troubleshoot. Indeed, the measure of a model or predictor is whether or not it provides an accurate estimation of the clinical signs of interest. If it does not, there is little indication as to why. The NLP practitioner may reason in terms of the statistical or numerical properties of the model, training procedure or data, but these arguments would not be grounded in a *psychological* theory. This is analogous to the level of detail in prediction: a coarser-grained assessment would be less operable than a finer-grained one. For example, a binary prediction for a disorder can hardly be challenged, still less corrected. A more detailed prediction describing a mental state in terms of signs and symptoms can be better approached in psychological language.

Although it would then seem that the solution is for the expected analyses to be as low-level as possible, psychology can only go so far before reaching its frontier of understanding of language use. In fact, this preoccupation is at the inception of language analysis in psychology. However, as psychology moved towards scientific methods, language analysis fell into disfavour, its qualitative means confining it to case-study use. Indeed, language being computationally inaccessible, it was difficult to devise more rigorous theory testing (Hevern, 2019). It would later gain resurgence in the form of statistics of distinguished word categories (Pennebaker et al., 2001; Stone & Hunt, 1963), under the hypothesis that a person's choice of words reveals their attention towards particular affective states. As discussed in Section 1.3, such measures discard important language information. Thus, this paradigm has become somewhat stagnant as it limits both what can be observed and what can be asked (Boyd & Schwartz, 2021). As NLP methods become more sophisticated, however, this is subject to change.

It is our position that future work in NLP and mental health should be predicated on building theories given the actual behaviour of NLP-based variables. That is, NLP methods should be treated as measures rather than as end-to-end problem solvers. This would allow for a sound, mutually informed nomological integration of these prospective factors with the clinical aspects of interest, as well as other explanatory factors of interest, such as cultural expression (Berger & Packard, 2022). We hope the present thesis will inspire work in this direction.

APPENDIX A
BECK DEPRESSION INVENTORY

Instructions:

This questionnaire consists of 21 groups of statements. Please read each group of statements carefully, and then pick out the one statement in each group that best describes the way you feel. If several statements in the group seem to apply equally well, choose the highest number for that group.

1. Sadness

    0. I do not feel sad.
    1. I feel sad much of the time.
    2. I am sad all the time.
    3. I am so sad or unhappy that I can't stand it.

2. Pessimism

    0. I am not discouraged about my future.
    1. I feel more discouraged about my future than I used to be.
    2. I do not expect things to work out for me.
    3. I feel my future is hopeless and will only get worse.

3. Past Failure

    0. I do not feel like a failure.
    1. I have failed more than I should have.
    2. As I look back, I see a lot of failures.
    3. I feel I am a total failure as a person.

4. Loss of Pleasure

    0. I get as much pleasure as I ever did from the things I enjoy.

1. I don’t enjoy things as much as I used to.
2. I get very little pleasure from the things I used to enjoy.
3. I can’t get any pleasure from the things I used to enjoy.

5. Guilty Feelings

   0. I don’t feel particularly guilty.
   1. I feel guilty over many things I have done or should have done.
   2. I feel quite guilty most of the time.
   3. I feel guilty all of the time.

6. Punishment Feelings

   0. I don’t feel I am being punished.
   1. I feel I may be punished.
   2. I expect to be punished.
   3. I feel I am being punished.

7. Self-Dislike

   0. I feel the same about myself as ever.
   1. I have lost confidence in myself.
   2. I am disappointed in myself.
   3. I dislike myself.

8. Self-Criticalness

   0. I don’t criticize or blame myself more than usual.
   1. I am more critical of myself than I used to be.
   2. I criticize myself for all of my faults.
   3. I blame myself for everything bad that happens.

9. Suicidal Thoughts or Wishes

   0. I don’t have any thoughts of killing myself.

    1. I have thoughts of killing myself, but I would not carry them out.
    2. I would like to kill myself.
    3. I would kill myself if I had the chance.

10. Crying

    0. I don’t cry anymore than I used to.
    1. I cry more than I used to.
    2. I cry over every little thing.
    3. I feel like crying, but I can’t.

11. Agitation

    0. I am no more restless or wound up than usual.
    1. I feel more restless or wound up than usual.
    2. I am so restless or agitated that it’s hard to stay still.
    3. I am so restless or agitated that I have to keep moving or doing something.

12. Loss of Interest

    0. I have not lost interest in other people or activities.
    1. I am less interested in other people or things than before.
    2. I have lost most of my interest in other people or things.
    3. It’s hard to get interested in anything.

13. Indecisiveness

    0. I make decisions about as well as ever.
    1. I find it more difficult to make decisions than usual.
    2. I have much greater difficulty in making decisions than I used to.
    3. I have trouble making any decisions.

14. Worthlessness

    0. I do not feel I am worthless.

1. I don’t consider myself as worthwhile and useful as I used to.

2. I feel more worthless as compared to other people.

3. I feel utterly worthless.

15. Loss of Energy

0. I have as much energy as ever.

1. I have less energy than I used to have.

2. I don’t have enough energy to do very much.

3. I don’t have enough energy to do anything.

16. Changes in Sleeping Pattern

0. I have not experienced any change in my sleeping pattern.

1a. I sleep somewhat more than usual.

1b. I sleep somewhat less than usual.

2a. I sleep a lot more than usual.

2b. I sleep a lot less than usual.

3a. I sleep most of the day.

3b. I wake up 1-2 hours early and can’t get back to sleep.

17. Irritability

0. I am no more irritable than usual.

1. I am more irritable than usual.

2. I am much more irritable than usual.

3. I am irritable all the time.

18. Changes in Appetite

0. I have not experienced any change in my appetite.

1a. My appetite is somewhat less than usual.

1b. My appetite is somewhat greater than usual.

2a. My appetite is much less than before.

- 2b. My appetite is much greater than usual.
- 3a. I have no appetite at all.
- 3b. I crave food all the time.

19. Concentration Difficulty

    0. I can concentrate as well as ever.
    1. I can't concentrate as well as usual.
    2. It's hard to keep my mind on anything for very long.
    3. I find I can't concentrate on anything.

20. Tiredness or Fatigue

    0. I am no more tired or fatigued than usual.
    1. I get more tired or fatigued more easily than usual.
    2. I am too tired or fatigued to do a lot of the things I used to do.
    3. I am too tired or fatigued to do most of the things I used to do.

21. Loss of Interest in Sex

    0. I have not noticed any recent change in my interest in sex.
    1. I am less interested in sex than I used to be.
    2. I am much less interested in sex now.
    3. I have lost interest in sex completely.

INDEX

BIBLIOGRAPHY

Abbott, M. W. (2020). The changing epidemiology of gambling disorder and gambling-related harm: Public health implications. *Public health.*

Abed-Esfahani, P., Howard, D., Maslej, M., Patel, S., Mann, V., Goegan, S. & French, L. (2019). Transfer learning for depression: Early detection and severity prediction from social media postings. In *CLEF (Working Notes).*

Achab, S., Chatton, A., Khan, R., Thorens, G., Penzenstadler, L., Zullino, D. & Khazaal, Y. (2014). Early detection of pathological gambling: betting on GPs' beliefs and attitudes. *BioMed research international*, *2014*.

Ageitos, E. C., Martínez-Romo, J. & Araujo, L. (2020). NLP-UNED at eRisk 2020: Self-harm early risk detection with sentiment analysis and linguistic features. In *Working Notes of the Conference and Labs of the Evaluation Forum-CEUR Workshop Proceedings*, volume 2696.

Aguirre, C., Harrigian, K. & Dredze, M. (2021). Gender and racial fairness in depression research using social media. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 2932–2949.

Altszyler, E., Berenstein, A. J., Milne, D. N., Calvo, R. A. & Slezak, D. F. (2018). Using contextual information for automatic triage of posts in a peer-support forum. In *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic*, pp. 57–68.

Ambalavanan, A. K., Jagtap, P. D., Adhya, S. & Devarakonda, M. (2019). Using contextual representations for suicide risk assessment from internet forums. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pp. 172–176.

American Psychiatry Association (2013). Highlights of changes from DSM-IV-TR to DSM-5.

Amir, S., Dredze, M. & Ayers, J. W. (2019). Health surveillance over social media with digital cohorts. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pp. 114–120. `https://doi.org/10.18653/v1/W19-3013`

Armstrong, M. D., Maupomé, D. & Meurs, M.-J. (2021). Topic modeling in embedding spaces for depression assessment. In *Proceedings of the Canadian Conference on Artificial Intelligence*. PubPub. https://caiac.pubpub.org/pub/b6tk9kak, `https://doi.org/10.21428/594757db.9e67a9f0`

Ba, J. L., Kiros, J. R. & Hinton, G. E. (2016). Layer normalization. *arXiv:1607.06450 [cs, stat]*. arXiv: 1607.06450, `http://arxiv.org/abs/1607.06450`

Bahdanau, D., Cho, K. & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473.*

Bai, J., Wang, Y., Chen, Y., Yang, Y., Bai, J., Yu, J. & Tong, Y. (2021). Syntax-BERT: Improving pre-trained Transformers with syntax trees. *arXiv:2103.04350 [cs]*. arXiv: 2103.04350, `http://arxiv.org/abs/2103.04350`

Baumgartner, J., Zannettou, S., Keegan, B., Squire, M. & Blackburn, J. (2020). The Pushshift Reddit dataset. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pp. 830–839.

Beck, A. T., Steer, R. A., Brown, G. K. et al. (1996). Beck depression inventory - second edition manual. *San Antonio, TX: Psychological Corporation*, *1*, 82.

Bell, C. C. (1994). DSM-IV: Diagnostic and statistical manual of mental disorders. *JAMA*, *272*(10), 828–829.

Beltagy, I., Peters, M. E. & Cohan, A. (2020). Longformer: The long-document transformer. *arXiv:2004.05150 [cs]*. arXiv: 2004.05150, `http://arxiv.org/abs/2004.05150`

Bengio, Y., Courville, A. & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, *35*(8), 1798–1828.

Berger, J. & Packard, G. (2022). Using natural language processing to understand people and culture. *American Psychologist*, *77*(4), 525.

Blei, D. M., Ng, A. Y. & Jordan, M. I. (2003). Latent Dirichlet allocation. *the Journal of machine Learning research*, *3*, 993–1022.

Bloom, D. E., Cafiero, E., Jané-Llopis, E., Abrahams-Gessel, S., Bloom, L. R., Fathima, S., Feigl, A. B., Gaziano, T., Hamandi, A., Mowafi, M. et al. (2012). *The Global Economic Burden of Noncommunicable Diseases*. Technical report, Program on the Global Demography of Aging.

Bojanowski, P., Grave, E., Joulin, A. & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146. `https://doi.org/10.1162/tacl_a_00051`

Boleda, G. (2020). Distributional semantics and linguistic theory. *Annual Review of Linguistics*, *6*, 213–234.

Bottou, L. et al. (1998). Online learning and stochastic approximations. *On-line learning in neural networks*, *17*(9), 142.

Bowman, S. R., Potts, C. & Manning, C. D. (2015). Recursive neural networks can learn logical semantics. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pp. 12–21., Beijing, China. Association for Computational Linguistics. `https://doi.org/10.18653/v1/W15-4002`

Boyd, R. L. & Schwartz, H. A. (2021). Natural language analysis and the psychology of verbal behavior: The past, present, and future states of the field. *Journal of Language and Social Psychology*, *40*(1), 21–41.

Braverman, J., Laplante, D., Nelson, S. & Shaffer, H. (2013). Using cross-game behavioral markers for early identification of high-risk internet gamblers. *Psychology of addictive behaviors : journal of the Society of Psychologists in Addictive Behaviors*, *27*, 868–77. `https://doi.org/10.1037/a0032818`

Burdisso, S. G., Errecalde, M. & Montes y Gómez, M. (2019). UNSL at eRisk 2019: a unified approach for anorexia, self-harm and depression detection in social media. In *Working Notes of the Conference and Labs of the Evaluation Forum-CEUR Workshop Proceedings*, volume 2380.

Canadian Mental Health Association (2020). Fast facts about mental illness.

Cao, K. & Rei, M. (2016). A joint model for word embedding and word morphology.

Cauchy, A. (1847). Méthode générale pour la résolution des systemes d'équations simultanées. *Comp. Rend. Sci. Paris*, *25*(1847), 536–538.

Cer, D., Yang, Y., Kong, S.-y., Hua, N., Limtiaco, N., John, R. S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C. et al. (2018). Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 169–174.

Chelba, C., Chen, M., Bapna, A. & Shazeer, N. (2020). Faster Transformer decoding: N-gram masked Self-Attention. *arXiv:2001.04589 [cs, stat]*. arXiv: 2001.04589, `http://arxiv.org/abs/2001.04589`

Cheng, J., Dong, L. & Lapata, M. (2016). Long short-term memory-networks for machine reading. *arXiv preprint 1601.06733*, *abs/1601.06733*. `http://arxiv.org/abs/1601.06733`

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., Belanger, D., Colwell, L. & Weller, A. (2021). Rethinking Attention with Performers. *arXiv:2009.14794 [cs, stat]*. arXiv: 2009.14794, `http://arxiv.org/abs/2009.14794`

Clark, K., Luong, M.-T., Le, Q. V. & Manning, C. D. (2020). ELECTRA: Pre-training text encoders as discriminators rather than generators.

Clevert, D.-A., Unterthiner, T. & Hochreiter, S. (2016). Fast and accurate deep network learning by exponential linear units (ELUs).

Coppersmith, G., Dredze, M. & Harman, C. (2014). Quantifying mental health signals in Twitter. In *Proceedings of the workshop on computational linguistics and clinical psychology: From linguistic signal to clinical reality*, pp. 51–60.

Coppersmith, G., Dredze, M., Harman, C., Hollingshead, K. & Mitchell, M. (2015). CLPsych 2015 shared task: Depression and PTSD on Twitter. In *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology (CLPsych): From Linguistic Signal to Clinical Reality*, pp. 31–39.

Cui, Y., Jia, M., Lin, T.-Y., Song, Y. & Belongie, S. (2019). Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9268–9277.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J. G., Le, Q. V. & Salakhutdinov, R. (2019). Transformer-XL: Attentive language models beyond a fixed-length context. *CoRR, abs/1901.02860*. http://arxiv.org/abs/1901.02860

Davidshofer, K. & Murphy, C. O. (2005). *Psychological Testing: Principles and Applications*. Pearson.

De Choudhury, M., Gamon, M., Counts, S. & Horvitz, E. (2013). Predicting depression via social media. In *Seventh international AAAI conference on weblogs and social media*.

Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dieng, A. B., Ruiz, F. J. & Blei, D. M. (2020). Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics*.

Dugas, C., Bengio, Y., Bélisle, F., Nadeau, C. & Garcia, R. (2001). Incorporating second-order functional knowledge for better option pricing. *Advances in neural information processing systems*, pp. 472–478.

Eichstaedt, J. C., Smith, R. J., Merchant, R. M., Ungar, L. H., Crutchley, P., Preoţiuc-Pietro, D., Asch, D. A. & Schwartz, H. A. (2018). Facebook language predicts depression in medical records. *Proceedings of the National Academy of Sciences*, *115*(44), 11203–11208.

Erhan, D., Courville, A., Bengio, Y. & Vincent, P. (2010). Why does unsupervised pre-training help deep learning? In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 201–208. JMLR Workshop and Conference Proceedings.

Ernala, S. K., Birnbaum, M. L., Candan, K. A., Rizvi, A. F., Sterling, W. A., Kane, J. M. & De Choudhury, M. (2019). Methodological gaps in predicting mental health states from social media: triangulating diagnostic signals. In *Proceedings of the 2019 chi conference on human factors in computing systems*, pp. 1–16.

Ferrone, L. & Zanzotto, F. M. (2020). Symbolic, distributed, and distributional representations for natural language processing in the era of deep learning: A survey. *Frontiers in Robotics and AI*, pp. 153.

Floridi, L. & Chiriatti, M. (2020). GPT-3: Its nature, scope, limits, and consequences. *Minds and Machines*, *30*(4), 681–694.

Gehring, J., Auli, M., Grangier, D., Yarats, D. & Dauphin, Y. N. (2017). Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1243–1252. JMLR. org.

Glorot, X. & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh & M. Titterington (eds.). *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 249–256., Chia Laguna Resort, Sardinia, Italy. PMLR. http://proceedings.mlr.press/v9/glorot10a.html

Glorot, X., Bordes, A. & Bengio, Y. (2011). Deep sparse rectifier neural networks. In G. Gordon, D. Dunson, & M. Dudík (eds.). *Proceedings of the Fourteenth International Conference on*

Artificial Intelligence and Statistics, volume 15 of *Proceedings of Machine Learning Research*, pp. 315–323., Fort Lauderdale, FL, USA. PMLR. http://proceedings.mlr.press/v15/glorot11a.html

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning.* MIT Press.

Grimsley, C., Mayfield, E. & R.S. Bursten, J. (2020). Why Attention is not explanation: Surgical intervention and causal reasoning about neural models. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pp. 1780–1790., Marseille, France. European Language Resources Association. https://www.aclweb.org/anthology/2020.lrec-1.220

Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D. & Smith, N. A. (2020). Don't stop pretraining: Adapt language models to domains and tasks. *arXiv:2004.10964 [cs].* arXiv: 2004.10964, http://arxiv.org/abs/2004.10964

Haefeli, J., Lischer, S. & Haeusler, J. (2015). Communications-based early detection of gambling-related problems in online gambling. *International Gambling Studies*, *15*(1), 23–38. https://doi.org/10.1080/14459795.2014.980297

Haefeli, J., Lischer, S. & Schwarz, J. (2011). Early detection items and responsible gambling features for online gambling. *International Gambling Studies*, *11*(3), 273–288.

Hahnloser, R. H. & Seung, H. S. (2001). Permitted and forbidden sets in symmetric threshold-linear networks. In *Advances in neural information processing systems*, pp. 217–223.

He, K., Zhang, X., Ren, S. & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV).*

Henderson, C., Evans-Lacko, S. & Thornicroft, G. (2013). Mental illness stigma, help seeking, and public health programs. *American Journal of Public Health*, *103*(5), 777–780.

Hermans, M. & Schrauwen, B. (2013). Training and analysing deep recurrent neural networks. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, & K. Q. Weinberger (eds.). *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2013/file/1ff8a7b5dc7a7d1f0ed65aaa29c04b1e-Paper.pdf

Hevern, V. W. (2019). The genesis of allport's 1942 use of personal documents in psychological science. *Qualitative Psychology*, *6*(1), 82.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, *abs/1207.0580.* http://arxiv.org/abs/1207.0580

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural networks*, *4*(2), 251–257.

Hu, M. & Liu, B. (2004). Mining and summarizing customer reviews. In *KDD'04: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM Press.

Ive, J., Gkotsis, G., Dutta, R., Stewart, R. & Velupillai, S. (2018). Hierarchical neural model with attention mechanisms for the classification of social media text related to mental health. In *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic*, pp. 69–77., New Orleans, LA. Association for Computational Linguistics. `https://doi.org/10.18653/v1/W18-0607`

Iyyer, M., Manjunatha, V., Boyd-Graber, J. & Daumé III, H. (2015). Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pp. 1681–1691.

James, S. L., Abate, D., Abate, K. H., Abay, S. M., Abbafati, C., Abbasi, N., Abbastabar, H., Abd-Allah, F., Abdela, J., Abdelalim, A. et al. (2018). Global, regional, and national incidence, prevalence, and years lived with disability for 354 diseases and injuries for 195 countries and territories, 1990–2017: A systematic analysis for the global burden of disease study 2017. *The Lancet*, *392*(10159), 1789–1858.

Ji, S. (2022). Private Correspondence.

Ji, S., Zhang, T., Ansari, L., Fu, J., Tiwari, P. & Cambria, E. (2022). MentalBERT: Publicly available pretrained language models for mental healthcare. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pp. 7184–7190.

Jordan, M. (1986). Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society (Erlbaum, Hillsdale, NJ), 1986.*

Jozefowicz, R., Vinyals, O., Schuster, M., Shazeer, N. & Wu, Y. (2016). Exploring the limits of language modeling. *arXiv preprint 1602.02410.*

Katharopoulos, A., Vyas, A., Pappas, N. & Fleuret, F. (2020). Transformers are RNNs: Fast autoregressive transformers with linear attention. In H. D. III & A. Singh (eds.). *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5156–5165. PMLR. `https://proceedings.mlr.press/v119/katharopoulos20a.html`

Kelleher, J. D., Mac Namee, B. & D'arcy, A. (2020). *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies.* MIT press.

Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, *abs/1412.6980.* `http://arxiv.org/abs/1412.6980`

Kitaev, N., Kaiser, Ł. & Levskaya, A. (2020). Reformer: The efficient Transformer. *arXiv:2001.04451 [cs, stat].* arXiv: 2001.04451, `http://arxiv.org/abs/2001.04451`

LeCun, Y. (1989). Generalization and network design strategies. *Connectionism in perspective*, *19*, 143–155.

Lin, T.-Y., Goyal, P., Girshick, R., He, K. & Dollár, P. (2018). Focal loss for dense object detection. *arXiv:1708.02002 [cs].*

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, *abs/1907.11692*. http://arxiv.org/abs/1907.11692

Losada, D. E., Crestani, F. & Parapar, J. (2018). Overview of eRisk – Early risk prediction on the Internet. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Ninth International Conference of the CLEF Association (CLEF 2018)*, Avignon, France.

Losada, D. E., Crestani, F. & Parapar, J. (2019). Overview of eRisk 2019: Early risk prediction on the Internet. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pp. 340–357. Springer.

Losada, D. E., Crestani, F. & Parapar, J. (2020). eRisk 2020: Self-harm and depression challenges. In *Advances in Information Retrieval*, pp. 557–563., Cham. Springer International Publishing. https://link.springer.com/chapter/10.1007/978-3-030-45442-5_72

Luitse, D. & Denkena, W. (2021). The great Transformer: Examining the role of large language models in the political economy of AI. *Big Data & Society*, *8*(2), 20539517211047734. Publisher: SAGE Publications Ltd, https://doi.org/10.1177/20539517211047734

Ma, Z. & Collins, M. (2018). Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3698–3707.

Maas, A., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y. & Potts, C. (2011). Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics*, pp. 142–150.

Madani, A., Boumahdi, F., Boukenaoui, A., Kritli, M. C. & Hentabli, H. (2020). USDB at eRisk 2020: Deep learning models to measure the severity of the signs of depression using reddit posts. In *CLEF (Working Notes)*.

Manning, C. & Schutze, H. (1999). *Foundations of Statistical Natural Language Processing.* MIT Press.

Martínez-Castaño, R., Htait, A., Azzopardi, L. & Moshfeghi, Y. (2020). Early risk detection of self-harm and depression severity using BERT-based transformers: iLab at CLEF eRisk 2020. In *CLEF (Working Notes)*.

Maupomé, D., Armstrong, M. D., Belbahar, R. M., Alezot, J., Balassiano, R., Queudot, M., Mosser, S. & Meurs, M.-J. (2020). Early mental health risk assessment through writing styles, topics and neural models. In *CLEF (Working Notes)*.

Maupomé, D. & Meurs, M. (2022). Contextualizer: Connecting the dots of context with second-order Attention. *Information.*, *13*(6), 290. https://doi.org/10.3390/info13060290

Maupomé, D. & Meurs, M.-J. (2018). Using topic extraction on social media content for the early detection of depression. In *CLEF (Working Notes)*, volume 2125.

Maupomé, D. & Meurs, M.-J. (2020). Language modeling with a general second-order RNN. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 4749–4753., Marseille, France. European Language Resources Association. `https://www.aclweb.org/anthology/2020.lrec-1.584`

Maupomé, D., Queudot, M. & Meurs, M.-J. (2019). Inter and intra document attention for depression risk assessment. In *Canadian Conference on Artificial Intelligence*, pp. 333–341. Springer.

Maupomé, D., Armstrong, M. D., Rancourt, F. & Meurs, M.-J. (2021a). Leveraging textual similarity to predict Beck Depression Inventory answers. *Proceedings of the Canadian Conference on Artificial Intelligence*. https://caiac.pubpub.org/pub/pkzbt8x2, `https://doi.org/10.21428/594757db.5c753c3d`

Maupomé, D., Fancourt, F., Armstrong, M. D. & Meurs, M.-J. (2021b). Position encoding schemes for linear aggregation of word sequences. *Proceedings of the Canadian Conference on Artificial Intelligence*. https://caiac.pubpub.org/pub/wvnuon1p, `https://doi.org/10.21428/594757db.37d7654d`

McGorry, P. D. & Mei, C. (2018). Early intervention in youth mental health: progress and future directions. *Evidence-based mental health*, *21*(4), 182–184.

Merchant, R. M., Asch, D. A., Crutchley, P., Ungar, L. H., Guntuku, S. C., Eichstaedt, J. C., Hill, S., Padrez, K., Smith, R. J. & Schwartz, H. A. (2019). Evaluating the predictability of medical conditions from social media posts. *PloS one*, *14*(6), e0215476.

Merity, S., Xiong, C., Bradbury, J. & Socher, R. (2016). Pointer sentinel mixture models. *arXiv:1609.07843 [cs]*. arXiv: 1609.07843, `http://arxiv.org/abs/1609.07843`

Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013). Efficient estimation of word representations in vector space.

Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J. & Khudanpur, S. (2010). Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

Oliveira, L. (2020). BioInfo@UAVR at eRisk 2020: On the use of psycholinguistic features and machine learning for the classification and quantification of mental diseases. In *CLEF (Working Notes)*.

Pang, B. & Lee, L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, pp. 271. Association for Computational Linguistics.

Pang, B. & Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pp. 115–124. Association for Computational Linguistics.

Parapar, J., Martin-Rodilla, P., Losada, D. E. & Crestani, F. (2021). Overview of eRisk 2021: Early risk prediction on the Internet. In *International Conference of the Cross-Language Evaluation Forum for European Languages*.

Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A. & Tran, D. (2018). Image Transformer. In J. Dy & A. Krause (eds.). *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4055–4064. PMLR. `https://proceedings.mlr.press/v80/parmar18a.html`

Pascanu, R., Mikolov, T. & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pp. 1310–1318. PMLR.

Pennebaker, J. W., Francis, M. E. & Booth, R. J. (2001). Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, *71*(2001), 2001.

Pennington, J., Socher, R. & Manning, C. D. (2014). GloVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Perotte, A., Wood, F., Elhadad, N. & Bartlett, N. (2011). Hierarchically supervised latent Dirichlet allocation. *Advances in neural information processing systems*, *24*, 2609–2617.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. & Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237.

Plank, B. (2011). *Domain adaptation for parsing.* (PhD thesis). University of Groningen.

Plank, B. (2016). What to do about non-standard (or non-canonical) language in nlp. *arXiv preprint.*

Qin, X., Wang, S. & Hsieh, C.-R. (2018). The Prevalence of Depression and Depressive Symptoms among Adults in China: Estimation Based on a National Household Survey. *China Economic Review*, *51*, 271–282.

Quillian, L., Pager, D., Hexel, O. & Midtbøen, A. H. (2017). Meta-analysis of field experiments shows no change in racial discrimination in hiring over time. *Proceedings of the National Academy of Sciences*, *114*(41), 10870–10875.

Rabanser, S., Shchur, O. & Günnemann, S. (2017). Introduction to tensor decompositions and their applications in machine learning. *arXiv preprint arXiv:1711.10781.*

Ravanbakhsh, S., Schneider, J. & Poczos, B. (2017). Equivariance through parameter-sharing. In *International conference on machine learning*, pp. 2892–2901. PMLR.

Resnik, P., Armstrong, W., Claudino, L., Nguyen, T., Nguyen, V.-A. & Boyd-Graber, J. (2015). Beyond LDA: Exploring supervised topic modeling for depression-related language in Twitter. In *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pp. 99–107.

Rosenblatt, F. (1957). *The perceptron, a perceiving and recognizing automaton Project Para.* Cornell Aeronautical Laboratory.

Roy, A., Saffar, M., Vaswani, A. & Grangier, D. (2020). Efficient content-based sparse Attention with routing Transformers. *arXiv:2003.05997 [cs, eess, stat].* arXiv: 2003.05997, `http://arxiv.org/abs/2003.05997`

Rude, S., Gortner, E.-M. & Pennebaker, J. (2004). Language use of depressed and depression-vulnerable college students. *Cognition & Emotion*, *18*(8), 1121–1133.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, *323*(6088), 533–536.

Sadeque, F., Xu, D. & Bethard, S. (2018). Measuring the latency of depression detection in social media. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 495–503.

Sánchez-Monedero, J., Dencik, L. & Edwards, L. (2020). What does it mean to 'solve' the problem of discrimination in hiring? social, technical and legal perspectives from the uk on automated hiring systems. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pp. 458–468.

Schotanus-Dijkstra, M., Drossaert, C. H., Pieterse, M. E., Boon, B., Walburg, J. A. & Bohlmeijer, E. T. (2017). An early intervention to promote well-being and flourishing and reduce anxiety and depression: A randomized controlled trial. *Internet Interventions*, *9*, 15–24. `https://doi.org/https://doi.org/10.1016/j.invent.2017.04.002`

Schuster, M. & Nakajima, K. (2012). Japanese and Korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5149–5152. ISSN: 2379-190X, `https://doi.org/10.1109/ICASSP.2012.6289079`

Schuster, M. & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, *45*(11), 2673–2681.

Schwartz, R., Dodge, J., Smith, N. A. & Etzioni, O. (2019). Green AI. *arXiv preprint arXiv:1907.10597*.

Sennrich, R., Haddow, B. & Birch, A. (2016). Neural machine translation of rare words with subword units. *arXiv preprint 508.07909*.

Shatz, I. (2017). Fast, free, and targeted: Reddit as a source for recruiting participants online. *Social Science Computer Review*, *35*(4), 537–549.

Shing, H.-C., Nair, S., Zirikly, A., Friedenberg, M., Daumé III, H. & Resnik, P. (2018). Expert, crowdsourced, and machine assessment of suicide risk via online postings. In *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic*, pp. 25–36., New Orleans, LA. Association for Computational Linguistics. `https://doi.org/10.18653/v1/W18-0603`

Shing, H.-C., Resnik, P. & Oard, D. W. (2020). A prioritization model for suicidality risk assessment. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8124–8137.

Socher, R., Manning, C. D. & Ng, A. Y. (2010). Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 deep learning and unsupervised feature learning workshop*, volume 2010, pp. 1–9. Vancouver.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C., Ng, A. & Potts, C. (2013). Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*.

Spärck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation.*

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929–1958.

Statistics Canada (2017). Accessing mental health care in Canada. `https://www150.statcan.gc.ca/n1/pub/11-627-m/11-627-m2017019-eng.htm`

Statistics Canada (2020). Care counts: Receiving care for a mental illness, 2018.

Stone, P. J. & Hunt, E. B. (1963). A computer approach to content analysis: studies using the general inquirer system. In *Proceedings of the May 21-23, 1963, spring joint computer conference*, pp. 241–256.

Strubell, E., Ganesh, A. & McCallum, A. (2019). Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3645–3650., Florence, Italy. Association for Computational Linguistics. `https://doi.org/10.18653/v1/P19-1355`

Substance Abuse and Mental Health Services Administration (2018). Key substance use and mental health indicators in the United States: Results from the 2017 National Survey on Drug Use and Health.

Sutskever, I., Martens, J. & Hinton, G. E. (2011). Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 1017–1024.

Teh, Y. W., Jordan, M. I., Beal, M. J. & Blei, D. M. (2006). Hierarchical Dirichlet processes. *Journal of the american statistical association*, *101*(476), 1566–1581.

Thomas, M. & Joy, A. T. (2006). *Elements of information theory.* Wiley-Interscience.

Trifan, A. & Oliveira, J. L. (2019). BioInfo@UAVR at eRisk 2019: Delving into social media texts for the early detection of mental and food disorders. In *CLEF (Working Notes).*

Uban, A.-S. & Rosso, P. (2020). Deep learning architectures and strategies for early detection of self-harm and depression level prediction. In *CLEF (Working Notes).*

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008.

Wahl, O. F. (2012). Stigma as a barrier to recovery from mental illness. *Trends in cognitive sciences*, *16*(1), 9–10.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O. & Bowman, S. (2019a). SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, pp. 3266–3280.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O. & Bowman, S. R. (2019b). GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv:1804.07461 [cs].* arXiv: 1804.07461, `http://arxiv.org/abs/1804.07461`

Widrow, B. (1960). *Adaptive ADALINE Neuron Using Chemical Memistors.* Technical report, Stanford University.

Wiebe, J., Wilson, T. & Cardie, C. (2005). Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, *39*(2), 165–210. https://doi.org/10.1007/s10579-005-7880-9

Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. *Neural computation*, *8*(7), 1341–1390.

World Health Organization (2013). *Mental Health Action Plan 2013-2020.* Technical report, World Health Organization.

Wu, H. C., Luk, R. W. P., Wong, K. F. & Kwok, K. L. (2008). Interpreting TF-IDF term weights as making relevance decisions. *ACM Transactions on Information Systems (TOIS)*, *26*(3), 1–37.

Yang, Z., Dai, Z., Salakhutdinov, R. & Cohen, W. W. (2018). Breaking the softmax bottleneck: A high-rank RNN language model. *arXiv preprint 1711.03953.*

Yates, A., Cohan, A. & Goharian, N. (2017). Depression and self-harm risk assessment in online forums. *CoRR*, *abs/1709.01848.* http://arxiv.org/abs/1709.01848

Younes, Z., Abdallah, F., Denoeux, T. & Snoussi, H. (2011). A dependent multilabel classification method derived from the k-Nearest Neighbor rule. *EURASIP Journal on Advances in Signal Processing*, *2011*, 1–14.

Zhang, M.-L. & Zhou, Z.-H. (2007). ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition*, *40*(7), 2038 – 2048.

Zirikly, A., Resnik, P., Uzuner, O. & Hollingshead, K. (2019). CLPsych 2019 shared task: Predicting the degree of suicide risk in Reddit posts. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pp. 24–33.