UNIVERSITÉ DU QUÉBEC À MONTRÉAL

TECHNIQUES D'APPRENTISSAGE PROFOND POUR LA MODÉLISATION DES USAGERS DANS LES SYSTÈMES INTERACTIFS D'APPRENTISSAGE HUMAIN

THÈSE
PRÉSENTÉE
COMME EXIGENCE PARTIELLE
DU DOCTORAT EN INFORMATIQUE

PAR
ANGE ADRIENNE NYAMEN TATO

MAI 2020

REMERCIEMENTS

Cette thèse n'aurait pas été possible sans le soutien et la contribution directe ou indirecte de plusieurs personnes à qui je voudrais témoigner toute ma gratitude.

Mes remerciements vont tout d'abord à mon directeur de thèse, Monsieur Roger Nkambou, professeur au Département d'informatique à l'UQAM, et à ma codirectrice de thèse, Madame Aude Dufresne, professeure au département de communication à l'UdeM. Je leur suis particulièrement reconnaissante pour leur soutien, leur disponibilité tout au long de ce travail ; leurs différentes remarques enrichissantes et leurs nombreux et précieux conseils.

Je remercie toute l'équipe du projet SoMoral, en particulier les professeurs Claude Frasson et Miriam Beauchamp de l'UdeM. Je remercie également toute l'équipe du projet Muse, en particulier le professeur Serge Robert de l'UQAM. Mes remerciements vont également au centre de recherche en intelligence artificielle (CRIA).

Je tiens à remercier tout particulièrement Madame Nathalie Guin, Maître de conférences HDR en informatique à l'Université de Lyon pour son aide si précieuse, et Juliette Laurendeau, qui m'a été d'une grande aide technique.

J'adresse mes plus sincères remerciements à mes parents, Mr Tato Eulalie et Mme Mbeunke Solange, mes frères et sœurs, Ariane, Joyce et Adam Tato, et mes proches qui de près ou de loin ont toujours su m'encourager et me soutenir dans mes travaux. Enfin, à mes amis et collègues Ghaith, Mickael, Pamela, Tan et Yacine, vont toute ma considération et ma reconnaissance pour leur support, leur collaboration et pour tous les bons moments partagés ensemble.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

RÉSUMÉ

La modélisation de l'usager dans les systèmes interactifs plus particulièrement dans les systèmes interactifs d'apprentissage humain, s'apparente à un processus de construction de sa représentation synthétique tel que vu par ces systèmes. On souhaite à terme, qu'au-delà de ce niveau de représentation, le modèle permette aussi d'établir certains liens de cause à effet et présente par conséquent une adéquation explicative. Cette modélisation consiste essentiellement en trois processus : (1) le traçage de modèle : savoir ce que fait l'usager, (2) la prédiction des actions de l'usager selon les connaissances qu'il possède, et (3) le traçage des connaissances : évaluer ce que l'usager a appris. Cette modélisation est généralement utilisée pour permettre (4) l'adaptation dans les systèmes interactifs.

Bien que la modélisation des usagers dans les systèmes interactifs soit centrale à leur efficacité, trop souvent elle est incomplète ou incertaine. Dans cette thèse, nous proposons une amélioration de la modélisation des processus (2), (3) et dans une certaine mesure (4) au moyen des techniques d'apprentissage profond. Ces techniques qui n'étaient jusqu'alors pas assez exploitées dans ce domaine, permettent d'extraire des représentations latentes utiles à la discrimination et à la prédiction. Nous proposons un cadre facilitant une modélisation précise de l'usager, et une approche d'adaptation simplifiée.

La première solution proposée est un algorithme permettant une optimisation de l'apprentissage dans les architectures profondes, afin de leur rendre propice à la modélisation de l'usager. La deuxième solution est une architecture hybride permettant de combiner les connaissances expertes et les données collectées, pour l'amélioration des modèles visant la prédiction de comportements. La troisième solution est une amélioration du traçage profond des connaissances (*Deep Knowledge Tracing*), qui prends en compte des compétences rares. La quatrième solution est une architecture multimodale permettant de fusionner intelligemment les différentes modalités (lorsque disponibles) du comportement de l'usager. Finalement la dernière solution est une technique d'extraction automatique des règles de production à l'aide d'un arbre de décision et d'un réseau de neurones, pour des fins d'adaptation. Toutes ces solutions ont été testées dans plusieurs cadres applicatifs, à l'instar d'un jeu sérieux pour l'apprentissage du raisonnement socio-moral.

Mots clés : Intelligence artificielle, apprentissage profond, modélisation de l'usager, systèmes d'apprentissage humain, système tutoriel intelligent, jeux sérieux.

INTRODUCTION

L'un des objectifs fondamentaux de la recherche sur l'interaction homme-machine est de rendre les systèmes interactifs plus attrayants, utiles et d'offrir aux utilisateurs des expériences qui leur permettent d'acquérir des connaissances de base liées à leurs objectifs spécifiques (Fischer, 2001). Les concepteurs de systèmes intelligents interactifs sont confrontés à la lourde tâche d'écrire des logiciels pour des millions d'utilisateurs tout en les faisant fonctionner comme s'ils étaient conçus pour chaque utilisateur individuel. Cette thèse se concentre sur le développement de solutions informatiques pour la modélisation des usagers. La modélisation des usagers est une étape importante pour la mise en place de systèmes interactifs adaptatifs (particulièrement ceux dédiés à l'apprentissage) capables de répondre convenablement aux besoins individuels des usagers qui peuvent êtres des apprenants, des joueur-apprenants ou encore simplement des joueurs.

0.1 Pourquoi la modélisation de l'usager ?

En tant que humains, nous avons cette capacité de nous construire inconsciemment une représentation latente d'une personne, rien qu'en observant ses faits et gestes. Cette représentation abstraite que fait notre cerveau par rapport à toute personne nous entourant nous permet par exemple de prédire ce qu'elle aimerait ou pas (pour le choix d'un cadeau d'anniversaire par exemple). C'est cette capacité qu'a notre cerveau de représenter une personne qu'on observe, que des chercheurs du domaine de la modélisation de l'usager (*User Modeling*) cherchent à implémenter dans un ordinateur.

Une partie importante de la recherche en Intelligence artificielle vise à «simuler l'intelligence humaine» (Russell et Norvig, 2016). Toutefois, malgré les avancées récentes des machines dites «intelligentes», leur capacité à se faire une représentation précise de l'humain même après l'avoir observé exécutants différentes tâches, reste très limiter. Cet écart entre la représentation que fait un humain d'un autre humain et la représentation que peut faire une machine d'un humain s'explique sans doute par le fait que, les algorithmes côté machine mis en place à cet effet, ne sont pas assez puissants pour non seulement bien exploiter les données disponibles sur l'usager mais également les exploiter dans leurs différentes perspectives. Cette thèse vise à pallier ce problème en proposant des techniques permettant d'améliorer les solutions actuelles de représentation et de modélisation du comportement humain à partir d'observations d'interactions avec un système. Une question évidente qu'on se poserait ici serait : pourquoi s'intéresser à la modélisation de l'humain par les machines ?

Dans le présent document, nous utilisons plus souvent les termes *apprenant* ou *joueur-apprenant*, bien que, pour des raisons de simplicité, le terme «*usager*» soit parfois utilisé. Ces termes sont utilisés pour référence à des usagers de systèmes d'apprentissage interactif humain. Le modèle de l'usager est une description abstraite de l'usager dans un environnement (Bakkes *et al.*, 2012). La modélisation d'un usager joue un rôle important dans les systèmes qui se veulent adaptatifs (Yannakakis *et al.*, 2013), entre autres les systèmes tutoriels intelligents (STI) (Nkambou *et al.*, 2010), les jeux sérieux (Michael et Chen, 2005), les MOOCs (*Massive Open Online Courses*), et même les applications et outils qui nous entourent au quotidien, à l'instar des réseaux sociaux et de nos téléphones mobiles. Il est donc important de développer une solution capable à partir de certaines observations, de modéliser l'usager fidèlement et d'être ainsi en mesure de prédire ses besoins, son comportement etc. Également, on aimerait avoir une solution qui à

partir des besoins prédits, soit capable de proposer des solutions pour répondre aux besoins spécifiques des usagers : capacité d'adaptation. Nous tenons à mentionner que cette problématique se rapproche de celle des systèmes de recommandations à l'exception que ces derniers utilisent généralement un profil utilisateur définit à la volée (attributs-valeurs) à partir de certaines actions qu'effectuent l'usager (Ricci *et al.*, 2015; Cheng *et al.*, 2016) alors que nous visons non pas un profil mais un modèle qui va représenter l'usager dans toutes ses facettes (modélisation multimodale). Dans cette thèse nous nous concentrons uniquement sur les systèmes interactif d'apprentissage humain à savoir les STI et les jeux sérieux, comme cadre applicatif. Mais nous pensons que les solutions proposées pourraient également servir dans d'autres domaines.

Un STI (Sleeman et Brown, 1982) est un système intelligent développé principalement pour l'aide à l'apprentissage avec comme support un ordinateur. Grâce à une modélisation interne de l'apprenant que le système se construit au fil des interactions, à une modélisation des connaissances du domaine qui est faite par des experts et à un module appliquant des règles tutorielles, il est capable de détecter les problèmes que rencontrent les étudiants et de leur apporter le soutien jugé nécessaire. Les STI opèrent généralement dans un environnement d'apprentissage interactif avec l'apprenant. Ces environnements peuvent prendre plusieurs formes et les jeux sérieux en sont un exemple. Ces jeux allient à l'apprentissage une dimension ludique et motivante, d'où les noms «sérieux» et «jeux». Il existe des solutions dans la littérature pour la modélisation des joueurs-apprenants dans ces systèmes. Cependant, ces solutions restent peu développées et peu efficaces en raison des techniques utilisées. De plus, elles produisent des modèles encore très peu fidèles aux usagers réels. Ces solutions existantes peuvent bénéficier des avancées récentes dans le domaine de l'apprentissage profond, également connu sous le terme *deep learning*) qui restent encore peu explorées dans ce domaine.

L'objectif de cette thèse est de développer de nouvelles techniques et des architectures profondes viasant à modéliser l'usager dans toutes ses facettes auquel le système a accès et à proposer des solutions semi-automatiques pour l'extraction de règles d'adaptation une fois la modélisation effectuée. L'importance de la multi-modalité ici, lorsque les données sont disponibles, est justifiée par le fait que l'usager ne peut pas être modélisé le plus fidèlement possible qu'à partir d'une seule modalité étant donné que le comportement humain en soi est multimodal (Lazarus, 1973). Par exemple, le fait de simplement écouter quelqu'un nous donne certes des informations sur cette personne mais en observant en plus ses faits et gestes, nous avons une information supplémentaire qui vient enrichir notre perception de lui, c'est à dire la représentation que fait le cerveau.

0.2 Problématique de recherche

La modélisation des usagers dans les systèmes intelligents est un sujet qui prend de plus en plus d'ampleur aujourd'hui. Ceci est dû notamment aux avancées dans le domaine des neurosciences où des outils sont de plus en plus disponibles pour permettre d'accéder à des données cognitives et psychologiques sur les usagers ou à les déduire et ce, sans passer par des techniques intrusives à l'instar des questionnaires pendant le déroulement de l'interaction. Ces données concernent entre autres la charge mentale, les émotions, le niveau d'attention, etc. Il devient nécessaire d'en tenir compte pour une modélisation fidèle de l'usager qui permettrait une personnalisation efficace des interactions.

Comme présenté dans la section précédente, il y a plusieurs avantages à développer des techniques capables de bien modéliser les usagers, entre autres pour mieux les comprendre et répondre à leurs besoins ou développer des systèmes adaptatifs plus performants. Intuitivement, un système dit adaptatif en est un

capable de modifier ses *artefacts* en temps réel ou en différé, en fonction des besoins de l'utilisateur courant. Les avantages d'un système adaptatif sont évidents puisqu'il permet d'adresser un public diversifié (Birk *et al.*, 2015). Ce n'est pas un concept nouveau en soi car bien avant les jeux sérieux, l'adaptation avait été largement considérée dans le domaine de l'AIED (*Artificial Intelligence In Education*) où l'un des objectifs est de rendre les systèmes tutoriels intelligents (STI) plus efficaces. L'adaptation du contenu et des interactions basées sur un modèle dynamique de l'apprenant reste l'une des principales questions traitées dans ce domaine (Woolf, 2010). L'efficacité des STI a été clairement démontrée mais leur utilisation à grande échelle n'est pour l'instant pas un succès, sauf dans certains secteurs comme la formation militaire, l'aérospatiale et le médicale. Des résultats d'études tendent à la conclusion que cet insuccès est principalement expliqué par le fait que les environnements d'apprentissage offerts ne sont pas suffisamment motivants et engageants pour les apprenants. Or, cette capacité de pouvoir attirer l'attention, exciter, offrir des défis et encourager la participation dans l'environnement est essentielle et contribue à motiver davantage l'usager dans sa tâche d'apprentissage (Sabourin et Lester, 2013).

Depuis quelques années, les recherches se sont orientées vers l'utilisation des jeux vidéo comme moyen de pallier ce manque. Cependant, créer un jeu qui peut à la fois éduquer et divertir n'est pas une tâche simple (Starks, 2014). Ces types de jeux doivent être capables de s'adapter au joueur-apprenant en tenant compte de tous ses attributs pertinents (Bontchev, 2016) : émotions, compétences, comportement social, etc. Plusieurs essais d'adaptation ont été mis en œuvre dans les jeux sérieux et les résultats ont montré que l'adaptation est plus efficace si elle est «*player-centered*» c'est-à-dire basée sur une modélisation du joueur-apprenant (Charles et Black, 2004; Charles *et al.*, 2005; Lopes et Bidarra, 2011). Toutefois, les fondements de l'adaptation, ainsi que les conditions et les effets de sa mise en

œuvre efficace et à grande échelle sont encore mal explorés (Reichart *et al.*, 2015). De ce constat, il en découle plusieurs questions importantes que nous présentons ci-dessous.

0.2.1 Comment concevoir une modélisation fidèle des usagers ?

Comment faut-il s'y prendre pour pouvoir modéliser les usagers de la façon la plus précise possible ? Le comportement humain est par nature multimodal. Dans certains systèmes, il est possible de capturer une ou plusieurs de ces modalités. Comment concevoir des modèles capables de représenter fidèlement chaque usager de systèmes intelligents à partir des différentes données capturées ?

Répondre à ces questions implique une avancée majeure non seulement dans le domaine des STI (ou de l'AIED) et des jeux sérieux, mais aussi dans la compréhension et la modélisation des comportements des usagers dans les systèmes interactifs en général. Les défis à relever sont donc les suivants :

1. Puisqu'il faudra prendre en considération un certain nombre de caractéristiques pertinentes de l'usager entre autres ses émotions, son comportement dans le jeu ou dans l'environnement interactif, ceci implique une manipulation de données provenant de diverses sources telles que les données relatives à l'historique des activités, les expressions faciales, les informations sur le regard, etc. Or la gestion de cette multiplicité de sources nécessite de trouver des techniques adéquates pour traiter ces données multimodales. Par conséquent, il est nécessaire d'investiguer et de mettre au point des algorithmes d'apprentissage machine multimodales efficaces qui permettront l'extraction d'attributs latents intéressants liés à ces caractéristiques ;
2. Dans plusieurs domaines en particulier dans le domaine de l'éducation, il existe plusieurs théories qui fournissent des indicateurs pouvant permettre

de parfaire la modélisation des usagers. Par exemple, des recherches ont déterminé l'importance du comportement non verbal dans le cadre d'une interaction. Comment tenir compte et intégrer de tels indicateurs dans la construction du modèle d'apprentissage machine ? Il nous faut à cet effet concevoir des solutions dites hybrides qui allient ces connaissances a priori et a posteriori aux données pour une modélisation plus performante ;

3. Les solutions visées doivent résoudre les problèmes de traçage des connaissances rares (connaissances difficiles/faciles à maîtriser) afin de permettre une meilleure généralisation sur ces types de connaissances ;
4. Les solutions envisagées doivent également être performantes au niveau du temps d'exécution, étant donné qu'elles seront utilisées pour des classifications et des prédictions en temps réel. Elles doivent donc être non seulement efficaces pour la modélisation de l'usager dans toute sa complexité, mais également produire des résultats en temps raisonnable pour une prise en considération en temps réel.

0.2.2 Comment mettre en place une adaptation efficace ?

Une fois un méta-modèle pour la production de modèles de l'usager mis en place, il convient de développer un moteur d'adaptation qui puisse les utiliser efficacement pour l'adaptation dans les systèmes interactifs. Les défis à relever pour la conception d'un moteur d'adaptation efficace sont nombreux. Tout d'abord, différentes techniques d'adaptation existent parmi lesquelles la génération automatique des contenus, l'adaptation selon les types d'utilisateur, l'adaptation du niveau de difficulté de l'activité ou alors l'ordre de déroulement des scènes ou des scénarios. Il s'agit d'explorer les techniques les plus prometteuses et de les raffiner afin qu'elles puissent tirer profit du modèle enrichi de l'usager qui sera développé. Ensuite, l'adaptation étant un processus qui est souvent effectué manuellement par les

experts du domaine, il est logique d'explorer des moyens pour rendre le développement d'un modèle d'adaptation automatique ou semi-automatique en utilisant les données récoltées par les systèmes. Dans cette perspective, une possibilité est d'extraire des règles d'adaptation à partir des données. Il faut alors identifier les attributs importants de l'usager (leviers – valeurs idéales qui peuvent avoir un impact positif sur l'apprentissage) par rapport à ceux non importants à observer et à modifier pour un apprentissage optimal. Enfin, les solutions proposées mériteront d'être transposables à plusieurs domaines afin d'élargir leur utilité à tout système interactif adaptatif aux besoins de l'usager. Cette exigence gouverne l'ensemble de nos choix méthodologiques tout au long du développement de nos solutions.

0.3 Hypothèses de la recherche

De la problématique de recherche décrite ci-dessus et des questions de recherche qui en découlent, nous avons formulé quelques hypothèses qui guideront notre travail :

- Les recherches dans le domaine de l'intelligence artificielle en éducation ont montré que l'intégration d'un modèle sophistiqué de l'utilisateur dans les environnements d'apprentissage, entraîne une adaptation plus flexible et efficace. Nous faisons l'hypothèse que ce constat est applicable dans tous systèmes interactifs d'apprentissage (jeux sérieux par exemple).
- Un modèle enrichi de l'usager qui inclut différents facteurs explicites (lorsque disponibles et captés par différentes modalités) liés aux compétences cognitives et méta-cognitives de celui-ci, à son profil affectif, à sa personnalité et son profil social, permet une adaptation réussie en terme d'efficacité pédagogique.
- Les architectures profondes ont la capacité d'extraire des informations latentes permettant d'abstraire assez fidèlement les données fournies en en-

trées et ainsi de les prédire et ou de les classifier. Nous faisons l'hypothèse qu'en nourrissant ces architectures de données d'utilisateurs, il est possible d'extraire des représentations latentes et assez précises de ces derniers.

- Le développement et/ou l'amélioration des techniques de fouille de données et d'apprentissage machine existantes favorisent une analyse efficace des données d'interaction multimodales et multi-sources (historique d'actions, expressions faciales, regard, verbatim de discours, etc.).

0.4 Objectifs de la recherche

L'objectif principal de cette thèse est de développer des algorithmes et des architectures de fouille de données et d'apprentissage profond pouvant servir de méta-modèle pour une modélisation plus fidèle du comportement humain dans les systèmes interactifs adaptatifs et pour une adaptation dont les règles sont extraites de manière semi-automatique. Cet objectif général se décline en 3 sous objectifs spécifiques que nous détaillons ci-dessous :

Objectif 1 : Développer de nouvelles techniques et des architectures pour la modélisation du comportement tant unimodal que multimodal :

- **Développer une nouvelle stratégie permettant d'accélérer la descente du gradient dans les algorithmes d'optimisation** : puisque nous visons une modélisation précise de l'usager, et qu'on aimerait que ce modèle se mette à jour le plus rapidement et le plus efficacement possible, nous proposons une nouvelle stratégie permettant d'accélérer l'apprentissage dans les architectures d'apprentissage profond et de trouver un meilleur modèle. De façon plus technique, la solution à développer permettrait au réseau de neurones de trouver un minimum d'une valeur plus basse pour la fonction d'erreur (une meilleure solution) en un temps plus

restreint comparé aux solutions existantes proposées dans l'état de l'art.

- **Développer une architecture de réseaux de neurones hybride tirant profit à la fois des données collectées et des connaissances a priori et a posteriori**. Dans plusieurs domaines, notamment en éducation et en médecine, les connaissances a priori et a posteriori qui ont été mises en place pendant des années voir des siècles, ne peuvent pas être ignorées au profit des données uniquement (*data-driven*). Pour cette raison, nous proposons une architecture qui combine données et connaissances a priori pour extraire une modélisation de l'usager capable de prédire ses comportements avec précision.
- **Développer une solution permettant de pallier le problème des comportements rares.** Étant donné que nous visons une approche qui implique tout de même l'utilisation des données, et que les modèles qui y sont extraits ne peuvent généraliser que sur les données qu'ils ont vues, il peut arriver que ces modèles ne soient par fidèles lors de modélisations de personnes ayant des comportements rares. En d'autres termes, s'il y a peu de données sur un comportement précis, les modèles appris ne seront pas capables de généraliser sur ce type de comportement étant donnée le peu d'exemples. S'il y a trop de données sur un comportement, les modèles appris auront tendances à associer tous les usagers à ce comportement. En utilisant les solutions visant à modifier la fonction d'erreur, nous proposons une technique permettant d'amplifier les comportements rares afin de pallier ce problème.
- **Développer une architecture multimodale** permettant d'extraire une représentation latente d'un usager (ex : joueur-apprenant) à partir des données multimodales recueillies.

Objectif 2 : Mettre en place des stratégies pour une adaptation efficace

- **Extraire les règles d'adaptation à partir des théories** ;

- **Extraire des règles d'adaptation à partir des données** : nous utilisons pour ce faire des techniques d'apprentissage machine telles que les arbres de décision et les réseaux de neurones ;
- **Utiliser le méta-modèle de l'usager pour l'adaptation** ;
- **Développer un moteur d'adaptation** : implémenter un moteur d'adaptation sous forme d'outil pour la définition automatique des règles d'adaptation dans un jeu sous *Unity 3D* (outil de développement de jeux 3D).

Objectif 3 : Valider par des expériences les solutions proposées dans des cadres applicatifs que nous avons développés.

- **Développer deux systèmes interactifs** dont l'un est un jeu sérieux et l'autre est un système tutoriel intelligent. Le premier système permet de recueillir des données multimodales alors que le deuxième ne donne accès qu'à une modalité.
- **Intégrer et tester les techniques et architectures de représentation du comportement humain dans les deux systèmes**. Il s'agit ici de valider les différents modèles de prédiction développés en particulier le modèle de prédiction des connaissances.
- **Intégrer et tester les solutions d'adaptation développées et les hypothèses spécifiques aux domaines d'application.** Il s'agit ici, de vérifier si l'apprentissage est effectif avec l'intégration de nos solutions comparé à un système ne contenant pas nos solutions. On évalue également la satisfaction des versions adaptatives de nos systèmes développés.
- **Intégrer et tester les techniques et architectures de représentation du comportement humain dans d'autres systèmes**. Il s'agit ici de valider les autres modèles de prédiction développés dans d'autres contextes et d'autres bases de données, autres que les deux systèmes développés par notre équipe.

0.5 Organisation de cette thèse

Ce document comporte 5 chapitres en plus de la conclusion.

Dans le premier chapitre — Fondements théoriques et état de l'art — nous donnons un aperçu des solutions existantes dans la littérature, les contextes dans lesquels elles sont employées et présentons les problèmes que rencontrent ces solutions. Nous détaillons l'état actuel de la recherche du point de vue de la modélisation *holistique* de l'usager. Nous montrons ensuite en quoi il est essentiel de résoudre ces problèmes en utilisant notamment l'apprentissage profond et en tenant compte de la multimodalité du comportement humain.

Dans le deuxième chapitre — Méthodologie et cadre expérimental — nous présentons la démarche méthodologie adoptée pour notre recherche. Dans ce chapitre, nous détaillons les théories et techniques existantes sur lesquelles reposent nos solutions, et comment nous nous y prenons pour les développer et les valider.

Dans le troisième chapitre — Une nouvelle technique d'accélération des algorithmes d'optimisation de 1er ordre — nous décrivons la première solution fondamentale que nous proposons pour l'accélération de l'apprentissage dans les architectures d'apprentissage profond. Cette description est accompagnée de plusieurs expérimentations sur des bases de données publiques, visant à évaluer son efficacité par rapport aux solutions d'optimisation classiques existantes.

Dans le quatrième chapitre — Nouvelles techniques et architectures pour la modélisation du joueur-apprenant — nous présentons les différentes solutions que nous proposons pour une meilleure modélisation de l'usager. Nous présentons notre solution hybride utilisant les connaissances à priori, notre solution permettant de résoudre les problèmes de connaissances rares, notre architecture multimodale et finalement nos techniques d'extraction automatique de règles d'adaptation.

Dans le cinquième et dernier chapitre — Applications des solutions et résultats — nous présentons des exemples d'applications intégrant nos solutions. Différents cadres applicatifs concrets développés par notre équipe pour tester l'efficacité de nos solutions sont décrits, mais également d'autres cadres applicatifs liés à d'autres projets de recherche. Nous y discutons aussi de la disponibilité de toutes les solutions proposées. Le chapitre se termine par une liste de nos publications en lien avec cette thèse.

Nous terminerons ce mémoire de thèse par une conclusion générale dans laquelle nous résumons nos contributions et leurs limites, nous discutons des hypothèses énoncées et présentons les possibles travaux futurs. Plus précisément, nous tentons de répondre aux nombreuses questions posées par nos approches, en identifiant tout d'abord leurs apports essentiels, ainsi que leurs limites. Nous affinons et généralisons par ailleurs quelques aspects de nos approches, sur la base des résultats que nous avons pu tirer de nos expérimentations. Pour finir, nous positionnons nos travaux par rapport aux contributions existantes et développons quelques perspectives.

La lecture de cette thèse ne devrait pas nécessiter de connaissances théoriques particulières en modélisation de l'usager. Une certaine base dans le développement ou l'utilisation d'architectures d'apprentissage profond est cependant requise pour aborder la plupart des discussions, en particulier dans le chapitre 3 et le chapitre 4. Ces deux chapitres ainsi que quelques annexes exigent des connaissances mathématiques et algorithmiques plus avancées. Le chapitre 1 et la conclusion, et dans une moindre mesure le chapitre 5, ne requièrent aucun pré-requis et suffisent à se faire une idée très générale du contexte et de nos approches.

CHAPITRE I

FONDEMENTS THÉORIQUES ET ÉTAT DE L'ART

Ce chapitre a pour but de présenter le cadre théorique de notre recherche. Tout d'abord, nous présentons les différentes approches de modélisation de l'usager existantes. Par la suite nous présentons des analyses critiques de l'état de l'art ainsi que des pistes de solutions.

1.1 Introduction

La modélisation de l'usager est une problématique se situant dans les recherches portant à la fois sur les interactions humains-machines (*human-computer interaction*), l'intelligence artificielle (IA), la psychologie, la philosophie, etc. Elle peut se définir comme étant l'étude des différents modèles computationnels ayant pour but d'obtenir une représentation (exploitable par la machine) de l'usager dans les systèmes intelligents hautement adaptatifs dont les jeux sérieux et les systèmes tutoriels intelligents (Farooq et Kim, 2016). Nous regroupons souvent ces systèmes sous l'expression «systèmes interactifs d'apprentissage humain» car, quelle que soit la forme qu'ils revêtent, ils visent généralement à faciliter le développement des compétences dans un domaine par un apprenant humain. Une telle initiative permet de détecter, de prédire, et d'exprimer le comportement et les sentiments des usagers menant à la personnalisation et à l'adaptation des systèmes inter-

actifs en fonction de leurs préférences, de leurs performances et de leurs besoins individuels.

Afin d'obtenir une adaptation efficace, le système doit pouvoir se faire une représentation des aspects importants à observer de ses usagers. Cette représentation est ce qu'on appelle communément un «modèle de l'usager» ou «modèle de l'utilisateur». À un certain moment, le modèle contient un cliché des caractéristiques de l'usager, telles que collectées, déduites et stockées par le système. Ce modèle est multi-facette et peut être créé soit automatiquement en utilisant des techniques d'intelligence artificielle comme la fouille de données ou l'apprentissage machine (*data-driven* ou *bottom-up*), soit dérivé des théories telles que celles sur le comportement, sur la cognition ou sur les émotions (*theory-driven* ou *top-down*) (Yannakakis *et al.*, 2013). Il est admis que toutes les facettes (ou dimensions) du modèle de l'usager peuvent être mesurées à partir de questionnaires préalables (modèle statique) ou de décisions et comportements de ce dernier durant les interactions (verbales et non-verbales) avec le système (modèle dynamique), incluant ses réactions physiologiques.

Le modèle usager est une représentation qui capture entre autres :

- les objectifs, les tâches et les buts : qu'est-ce que l'usager essaye de faire ?
- les connaissances, les antécédents et l'expérience : que sait l'utilisateur du sujet ? que peut-on s'attendre à ce que l'utilisateur sache ?
- les centres d'intérêt : qu'est-ce que l'utilisateur aime ?
- les traits : traits de personnalité qui peuvent influencer le comportement et les attentes de l'utilisateur - par exemple, styles introvertis ou extravertis - cognitifs : holiste ou sérialisé.
- le contexte du travail : plate-forme, lieu, activité.

Il est à noter que ces éléments peuvent figurer tous ou en partie dans un modèle quelconque. Le nombre ou le type d'éléments pouvant y figurer dépend du

contexte et de l'importance que peut avoir chacun sur une bonne modélisation. Le profil de l'usager est un terme étroitement lié à la modélisation de l'usager et qui prête souvent à confusion. Le profil de l'usager contient la plupart du temps des informations spécifiées par l'utilisateur par opposition aux informations capturées sur l'utilisateur. Il peut aussi être vu comme une simplification de l'expression «modèle de l'utilisateur», plus lourde d'un point de vue sémantique. Nous pouvons grouper les modèles usagers en deux principales catégories : les modèles empiriques et les modèles analytiques. Dans les modèles empiriques, la modélisation est basée sur l'observation du comportement empirique de l'usager dont son interaction avec le système, l'objectif n'étant pas de comprendre son processus cognitif. Dans cette catégorie nous avons les modèles basés sur des stéréotypes et des modèles basés sur des attributs (*feature-based models*) de l'usager tel que les traits de personnalité. Dans les modèles analytiques, la modélisation vise à simuler le processus cognitif qui est mis en jeu durant les interactions de l'usager avec le système. Les modèles analytiques peuvent aussi être fusionnés avec les modèles empiriques.

La modélisation de l'usager permet de cibler une adaptation du système qui répond aux besoins individuels, une nécessité qui pourra grandement contribuer à l'efficacité du système. Toutefois, tenir compte du modèle de l'usager implique de déterminer les paramètres pertinents qui le caractérisent. Plusieurs questions sont alors soulevées (Brisson *et al.*, 2012). Quelles sont les composantes pertinentes d'un modèle de l'usager ? Quelles mesures ou caractéristiques doit-on considérer afin d'estimer en temps réel l'état actuel de l'usager dans chacune de ces composantes ? Quelles techniques peut-on utiliser ou développer pour inférer sur les objectifs/buts des utilisateurs ? Comment mettre en place une adaptation efficace sachant un modèle usager conçu ? Les travaux que nous présentons tout au long de ce chapitre tentent d'apporter quelques éléments de réponse à ces questions.

Dans la suite de ce chapitre, nous nous concentrons majoritairement sur la modélisation de l'usager dans les jeux puisque les mêmes enjeux concernent les STI et les autres systèmes interactifs. De plus, la modélisation de l'usager dans les jeux ainsi que les techniques utilisées sont largement inspirées des travaux effectués dans le domaine des STIs, et sont beaucoup plus récentes.

1.1.1 La modélisation de l'usager : état des lieux

Il existe 3 approches pour la modélisation de l'usager : explicite, implicite et hybride (Frias-Martinez *et al.*, 2009). Dans l'approche explicite, le modèle est construit à partir d'inférences humaines mais peut aussi être construit à l'aide de l'usager lui même. On demande directement à l'utilisateur quels sont ses états pendant le déroulement des interactions avec le système. Ses caractéristiques sont renseignées par lui-même — exemple : ses préférences, ses hobbies, etc. —. Cette approche est connue pour avoir des limites en matière d'évolutivité (*scalability*) et d'extensibilité. Les approches statistiques implicites sont plus souples et mieux adaptées au traitement de grandes quantités de données. Les modèles d'usagers implicites sont construits à partir des données brutes des utilisateurs, qui constituent l'entrée pour les mécanismes d'adaptation. Les inconvénients de cette approche sont que (1) les inférences produites à partir des données récoltées peuvent être inexactes ; (2) elle nécessite le plus souvent les connaissances d'un expert. De plus le processus implicite n'est pas intuitif comme le processus explicite et les modèles qui en ressortent ne sont parfois pas interprétables. L'approche hybride quant à elle est une combinaison de ces 2 approches.

La conception d'un modèle usager se fait en plusieurs étapes :

- Le choix du domaine : dans quelles tâches devront nous accompagner (assister) les usagers ?

- La création du modèle : explicite ou implicite. Dans l'explicite, les utilisateurs sont observés et les connaissances expertes sont utilisées pour construire un modèle cognitif de ces utilisateurs. Dans la création implicite, les données des utilisateurs sont recueillies et les approches d'apprentissage automatique sont utilisées pour extraire les modèles ;
- L'utilisation du modèle et l'adaptation ;
- L'évaluation.

Des alternatives à la modélisation de l'usager qui sont également très populaires sont la modélisation collaborative de l'usager (*Collaborative User Modeling* (Kim *et al.*, 2011; Ekstrand *et al.*, 2011)) — les usagers sont modélisés en fonction des autres — et la modélisation basée sur les préférences (*Preference-based User Modeling* (Fridgen *et al.*, 2000)).

La modélisation implique l'estimation en temps réel des caractéristiques de l'usager. Traditionnellement, les caractéristiques des utilisateurs qui sont le plus souvent prises en compte sont (Rich, 1979; Kobsa, 1993; Kobsa, 2001) (1) les informations démographiques (Fink et Kobsa, 2002; Garcia *et al.*, 2011) : des données démographiques simples peuvent être utilisées pour un réglage initial de l'interface, par exemple la localisation ; (2) les objectifs et les tâches de l'utilisateur (McCrickard et Chewar, 2003) : ils sont utilisés pour satisfaire les besoins de l'utilisateur de la manière la plus efficace et la plus efficiente possible ; (3) les connaissances de base de l'utilisateur (Razmerita *et al.*, 2003; Martins *et al.*, 2008) : elles sont utiles pour déterminer quels sont les concepts qu'un utilisateur connaît déjà et ceux qui nécessitent des explications supplémentaires ; (4) les intérêts des utilisateurs : ils sont utilisés pour déterminer l'information, les services ou les produits que les utilisateurs sont les plus susceptibles d'apprécier ; (5) les traits de caractère de l'utilisateur : comme les facteurs de personnalité, les facteurs cognitifs et les styles d'apprentissage (Ortigosa *et al.*, 2014) ; (6) l'humeur de l'utilisateur : est-il

heureux, stressé, détendu, tendu, effrayé, motivé, ennuyé, engagé, confu, frustré ; etc. (Qian *et al.*, 2019). Il est à noter que certaines caractéristiques sont plus faciles à obtenir ou à déduire que d'autres. Ces caractéristiques appartiennent à des dimensions différentes du modèle usager.

En effet, un modèle de l'usager efficace doit être multi-facette (multi-dimension) ; ceci s'explique par le fait que le comportement humain en soi est multidimensionnel (Levinson, 2006) (on parlera également de multi-modalité parce que chaque dimension provient généralement de canaux différents). Comme le mentionne Lazarus (Lazarus, 1973) (version traduite de l'anglais) :

> «Les humains sont des organismes biologiques (entités neurophysiologiques/ biochimiques) qui se comportent (agissent et réagissent), éprouvent des émotions (ressentent des réactions affectives), ressentent (répondent à des stimuli tactiles, olfactifs, gustatifs, visuels et auditifs), imaginent (évoquent des images, sons et autres événements dans notre esprit), pensent (entretiennent des convictions, opinions, valeurs et attitudes), et interagissent les uns avec les autres (aiment, tolèrent ou souffrent de diverses relations interpersonnelles).»

Ainsi, les caractéristiques telles que le mouvement des yeux, la gestuelle, la parole, etc. sont toutes des dimensions (pouvant être capturées à partir de senseurs) qui peuvent aider à mieux représenter l'usager humain dans sa globalité. Il est admis que toutes les facettes (ou dimensions) peuvent être mesurées à partir de questionnaires préalables ou de décisions et comportements de l'usager durant l'interaction avec le système, incluant ses réactions physiologiques. Nous allons nous intéresser aux deux principales facettes les plus populaires dans le développement de systèmes interactifs pour l'apprentissage à savoir : l'état cognitif en particulier le traçage de connaissances et l'état émotionnel.

1.1.1.1 État cognitif

L'état cognitif fait référence au processus cognitif mis en œuvre par l'usager dans l'exécution de ses actions. Elle fait aussi référence au processus d'acquisition des connaissances de l'usager. Il y a également le volet métacognitif dont nous ne parlons pas dans le cadre de cette thèse. La facette cognitive est modélisée par des architectures cognitives telles que EPIC (*Executive-Process Interactive Control*) (Meyer et Kieras, 1997) ou l'ACT-R (*Adaptive Character of Thought*) d'Anderson (Anderson *et al.*, 1997), capables d'analyser le comportement d'un apprenant et d'inférer sur son état cognitif, cet état étant généralement latent et déductible à partir des performances observables (exemple : réponse à un exercice). La modélisation des connaissances en anglais *Knowledge Tracing*, est probablement la partie de la modélisation du processus cognitif la plus étudiée et la plus présente dans les systèmes interactifs visant l'apprentissage. Elle consiste à modéliser l'évolution de l'état latent des connaissances au cours d'activités visant l'acquisition des compétences. Les connaissances de l'usager peuvent être représentées sous de nombreux formats différents.

- **Le modèle plat** (Henze et Herder, 2011) : Le modèle de base est un ensemble de variables simples et de valeurs associées (souvent représenté par un tableau à 2 dimensions). Ces variables peuvent représenter une variété de caractéristiques indépendantes, comme les caractéristiques démographiques de l'utilisateur, le goût de certains éléments de l'interface et les connaissances sur certains sujets.
- **Le modèle hiérarchique** (Ahmed *et al.*, 2013; Sreedharan *et al.*, 2018; Wu *et al.*, 2019) : Cette solution permet de considérer certains aspects du modèle usager comme étant de niveau supérieur et plus général que d'autres. Contrairement au modèle plat, les structures hiérarchiques représentent les caractéristiques des utilisateurs et les relations entre ces carac-

téristiques. Une structure hiérarchique commune est une arborescence ou un graphique acyclique dirigé. Les hiérarchies sont établies à la main sur la base des connaissances du concepteur.

- **Le stéréotypage** (Rich, 1979) : Un stéréotype représente un ensemble d'attributs qui sont souvent présents de manière concurrente chez des personnes (Rich, 1989). Ils permettent de grouper les usagers en fonction de certaines classes, par exemple l'expertise (ex : novice, intermédiaire, expert, etc.). Ces groupes sont spécifiés à l'avance, manuellement ou extraits à partir des données. Les stéréotypes sont particulièrement utiles lorsqu'on dispose d'une grande quantité de données statistique sur les groupes d'usagers. Ils sont également utiles pour déduire rapidement le type d'usager avec lequel un système interagi, dans le but de fournir des adaptations spécifiques à ce type d'utilisateur.
- **La superposition** (Carr et Goldstein, 1977; Brusilovsky et Millán, 2007) : Le modèle usager peut être vu comme une superposition de la structure du domaine. En d'autres termes, les connaissances de l'usager sont considérées comme formant un sous-ensemble de celles de l'expert. Ce type d'approche de modélisation de l'apprenant peut être relativement simple quand il s'agit de déterminer, sans diagnostic, si l'état de connaissance de l'apprenant correspond à la représentation de l'expert ; il peut aussi se complexifier si l'on vise une superposition du modèle du novice à celui de l'expert, de façon à pouvoir diagnostiquer beaucoup plus spécifiquement les déviations de l'usager par rapport au modèle visé par l'apprentissage (Giardina et Laurier, 1999).

1.1.1.2 État affectif

L'état affectif qui peut être à court terme (les émotions) ou à long terme (motivation, humeur et profil psychologique) fait référence aux réactions émotionnelles que peuvent ressentir les usagers lors de l'interaction avec le système. Ces réactions émotionnelles ont une influence capitale sur leurs aptitudes cognitives, comme la perception et la prise de décision (Jraidi, 2014). La modélisation des émotions est particulièrement pertinente pour les systèmes interactifs d'apprentissage humain. Ces systèmes cherchent à identifier les émotions de l'usager lors des sessions d'apprentissage, et à optimiser son expérience d'interaction en recourant à diverses stratégies d'intervention. Il existe plusieurs approches pour la modélisation des émotions dont celles qui nous intéressent qui sont les approches discrètes et les approches continues. Dans les approches discrètes, on considère que les émotions de base sont en nombre fini (nombre qui varie selon la théorie), énuméré par des théories telles que celles de Ortony (Ortony *et al.*, 1990) (modèle *OCC* des émotions) et Ekmann (Ekman, 1999) (études des émotions à partir des expressions faciales). Le modèle OCC des émotions est un modèle d'émotion largement utilisé qui affirme que la force d'une émotion donnée dépend principalement des événements, agents ou objets dans l'environnement de l'agent présentant l'émotion. Le modèle spécifie environ 22 catégories d'émotions et se compose de cinq processus qui définissent le système complet. Ces processus sont notamment : a) la classification de l'événement, de l'action ou de l'objet rencontré, b) la quantification de l'intensité des émotions affectées, c) l'interaction de l'émotion nouvellement générée avec les émotions existantes, d) le *mappage* de l'état émotionnel à une expression émotionnelle et e) l'expression de cet état émotionnel. Dans le modèle de Paul Ekman, les expressions faciales correspondent à 6 émotions fondamentales universellement connues : la colère, la peur, la tristesse, la joie, le dégoût et la surprise. Il étendra par la suite ces six émotions essentielles avec d'autres émotions

telles que la honte et le mépris. Les autres émotions en sont des nuances. Par exemple, l'indignation - l'exaspération - le tracas sont des teintes de la «colère» ; le bonheur - le soulagement - l'euphorie des teintes de la joie. Une hypothèse forte gouverne les approches discrètes : chacune des émotions se manifeste de la même manière sur tous les êtres vivants. Dans les approches continues, les émotions sont en nombre infini. Une émotion est en fait un point dans l'espace dont les dimensions les plus utilisées sont la valence et l'excitation encore appelé l'*arousal* en anglais (Barrett, 1998). La valence mesure la polarité de l'émotion qui peut être positive, négative ou neutre alors que l'*arousal* mesure le niveau d'excitation ou encore l'intensité de l'émotion. Contrairement aux approches discrètes, les approches continues ne permettent pas de donner un sens clair aux émotions, et la distinction entre certaines émotions peut devenir ambiguë.

1.1.2 Les enjeux et les défis d'une modélisation de l'usager dans les systèmes interactifs d'apprentissage humain

Trois tâches peuvent être séparées dans le processus de modélisation d'un usager apprenant :

- l'acquisition des données/caractéristiques de l'utilisateur ;
- l'inférence des connaissances à partir des données ;
- et la représentation du modèle utilisateur.

Ces tâches sont difficiles à mettre en œuvre dans des cas pratiques en raison principalement de la variabilité du contexte applicatif. Nous explorons chacune de ces tâches dans les sous-sections qui vont suivre.

1.1.2.1 Quelles caractéristiques faudrait-il considérer et comment les capturer ?

Les données sur l'usager sont constituées d'événements et d'observations sur son interaction avec le système, qui peuvent être soit directement utilisées à des fins

d'adaptation, soit prétraitées en fonction des caractéristiques de l'utilisateur sélectionnées au préalable. Quelles caractéristiques pertinentes faudrait-il prendre en considération ? pourquoi ? et comment prétraiter les données brutes afin d'extraire de telles caractéristiques ?

Les caractéristiques de l'usager peuvent être divisées en 2 groupes : les caractéristiques dépendantes du domaine et celles indépendantes du domaine (Benyon et Murray, 1993; Martins *et al.*, 2008). Les caractéristiques indépendantes du domaine sont celles provenant du profil générique dont les domaines d'intérêt, et du profil psychologique dont le profil cognitif et le profil affectif. Ces caractéristiques ne dépendent pas du domaine implémenté dans le système interactif et sont plus faciles à déterminer. Les caractéristiques dépendantes du domaine dont les objectifs de l'usager, ses connaissances actuelles en lien avec le domaine en jeu, ses connaissances du domaine inférées par le système, quant à elles, sont plus difficiles à identifier et à capturer car elles dépendent du contexte applicatif.

Les caractéristiques liées au profil de l'usager, sont souvent recueillies directement à partir des renseignements qu'il fournit. Elles peuvent également être recueillis pendant que l'utilisateur interagit avec le système. Dans les systèmes de recommandation, la rétroaction est un élément essentiel du système, car le processus de recommandation repose principalement sur les évaluations des utilisateurs et les critiques des éléments du système (produits, films, etc.) (He *et al.*, 2017). Dans de nombreux cas, les usagers veulent simplement commencer à travailler et interagir avec le système sur leurs tâches sans avoir à lire des manuels, à suivre une visite d'introduction ou à remplir des formulaires. Ainsi, de nombreux systèmes adaptatifs tendent de déduire directement les connaissances en surveillant discrètement les interactions de l'utilisateur avec le système.

Sélectionner les caractéristiques intéressantes pour une modélisation de l'usager

est une tâche qui relève du concepteur du système. Il n'existe pas de solutions génériques à ce jour pour ce problème. Un modèle idéal intégrerait toutes les caractéristiques possibles. Cependant, il n'est pas possible techniquement de capturer toutes les facettes et les caractéristiques associées de l'usager dans un seul système car il y a des problèmes d'équipements, de temps de traitement et de pertinence. Les concepteurs de systèmes interactifs se basent alors sur les théories, les travaux antérieurs reliés à leur contexte et les équipements à leur disposition (senseurs, etc.) pour la sélection et l'acquisition des caractéristiques des usagers.

Une fois les caractéristiques sélectionnées, il faudrait être capable de les acquérir durant les sessions d'interactions. Pour les caractéristiques liées à l'état cognitif, on utilisera les données d'un journal d'activités ou historique dont le format et le contenu doivent être prévus en conséquence, alors que pour l'état affectif, on utilisera les données physiologiques provenant des senseurs ou équipements spécifiques tels que le Tobii pour le traçage du mouvement oculaire (Duchowski, 2007), le Facereader [1](Noldus, 2014) pour la reconnaissance des émotions à partir des expressions faciales ou encore les casques EEG (ElectroEncephaloGram) pour la mesure l'activité électrique du cerveau à partir des électrodes placées sur le cuir chevelu.

Un autre défi ici est de pouvoir déterminer le moment opportun pour mettre à jour les caractéristiques des utilisateurs, et dans quelles situations il faudrait le faire. De plus, il y a une certaine incertitude à gérer car toutes les observations ne peuvent pas toujours être traduites en caractéristiques de l'usager avec une confiance de 100%.

---

1. www.noldus.com

1.1.2.2 Comment inférer les caractéristiques à partir des données ?

Une fois les données acquises et les caractéristiques utiles sélectionnées avec une certaine précision, il faut mettre en place des techniques permettant l'extraction et l'interprétation des valeurs de ces caractéristiques pour la modélisation. Par exemple, comment inférer sur le processus cognitif d'un usager à partir de son patron de comportements dans le jeu : réponses aux exercices, temps de réponse, etc ? L'inférence de connaissances est le processus d'interprétation des événements et des observations sur un usager, qui peut supposer l'utilisation de conditions, de règles ou d'autres formes de raisonnement, et le stockage des connaissances inférées dans le modèle utilisateur. De nombreuses interactions ont un sens en soi, telles que les visites de pages, les actions de *bookmarking* ou d'enregistrement, les requêtes émises par l'utilisateur et les articles inspectés ou achetés sur un site Web de commerce électronique. D'autres interactions doivent être combinées ou interprétées afin de devenir significatives, comme les touches, les clics de souris et le comportement du regard. L'hypothèse de l'inférence des connaissances est que l'interaction de l'utilisateur avec un système peut être prédite dans une certaine mesure. Encore là, il n'existe pas de solutions universelles ; il revient au concepteur de choisir ou de développer l'architecture appropriée, par exemple un réseau bayésien (Jensen *et al.*, 1996) pour l'extraction du processus d'acquisition des compétences. Il faut également considérer que la modélisation des différentes caractéristiques de l'utilisateur peut se faire en même temps. C'est encore plus compliqué lorsque les sources de données ne sont pas structurées. De telles données nécessitent quelques actions de nettoyage et d'analyse pour extraire l'information requise.

1.1.2.3 Comment représenter l'usager dans le système ?

Le modèle usager est une structure de données qui caractérise un usager quelconque à un instant donné dans un système interactif. Les données capturées par le système interactif peuvent être représentées sous plusieurs formats dans le modèle usager. Il est possible d'utiliser pratiquement n'importe quel format à l'instar des couples attribut valeur, des intervalles de probabilité, des booléens, des intervalles flous, des listes éventuellement avec poids, des règles, des heuristiques, des références à des objets externes, des ontologies (Zhang *et al.*, 2007), des réseaux de neurones, etc. De plus, des méta-données peuvent être utilisées pour déduire l'origine des données, les contextes dans lesquels elles sont valides, le moment où il faut les utiliser, etc. Il n'existe pas de représentation universelle, le défi est donc de trouver celle qui se prête le mieux au contexte.

1.1.3 Les défis d'un système adaptatif

Une analyse profonde des travaux de recherche montre que deux grands défis sont à relever pour tout système qui se veut adaptatif au modèle de l'usager : la reconnaissance du comportement de l'usager et l'adaptation du contenu à un état de ce comportement et selon les objectifs de l'usage (ex : objectifs d'apprentissages) (Brisson *et al.*, 2012). Le modèle usager est conçu principalement dans le but d'adapter le système interactif aux besoins de ses usagers. Ainsi, le modèle usager est consulté périodiquement par le système pour déterminer le focus de l'interaction (ex : l'apprentissage humain dans le cas des STI). Le processus d'adaptation, inspiré des travaux de (Paramythis *et al.*, 2010), se résume dans la figure 1.1. Comme présenté dans cette figure, le processus d'adaptation commence par une collecte des données sur l'usager et une interprétation de ces données à partir de modèles statiques ou dynamiques. Ensuite, cette interprétation permet de modé-

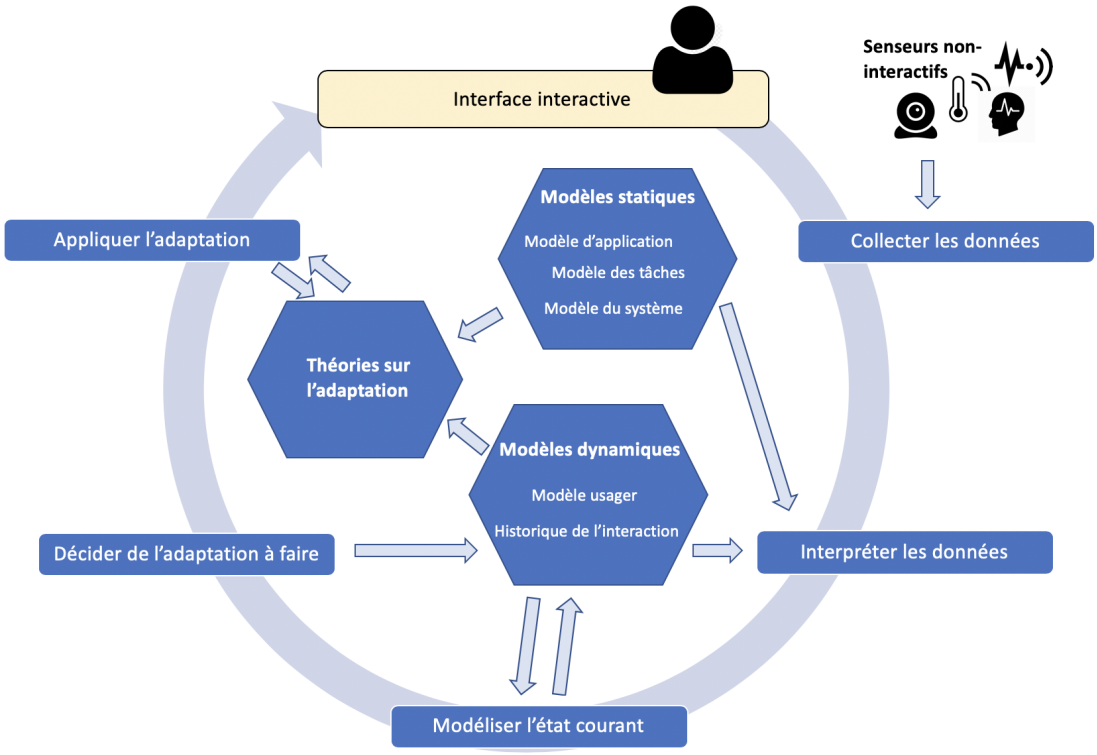


Figure 1.1 Processus d'adaptation dans les systèmes adaptatifs interactifs (Paramythis *et al.*, 2010)

liser l'état courant de l'usager qui sert donc à l'adaptation. L'adaptation peut se faire par l'usager lui-même en commandant des listes, en activant ou désactivant des options, en faisant glisser des éléments d'interface ou par toute autre interaction spécifique avec le système. Elle peut également se faire de façon automatique, implicitement, c'est-à-dire que les procédures d'adaptation sont inséparables du code du système, ou explicitement, c'est-à-dire que les procédures d'adaptation se fondent sur des modèles explicites tels qu'un moteur de règles ou un réseau de neurones. L'adaptation en soi peut concerner autant la présentation que le contrôle ou le contenu. Une adaptation de présentation consiste à adapter l'interface du système interactif par exemple, les feedbacks dans le cadre d'un jeu. Une adaptation de contrôle consiste à adapter les règles du système par exemple, modifier

la difficulté d'un jeu, modifier les paramètres d'un personnage non joueur. Une adaptation de contenu consiste à adapter le contenu en tant que tel par exemple, adaptation ou génération automatique de scénarios de jeu ou d'apprentissage. Comme présenté à la figure 1.1, l'adaptation se base principalement sur les théories existantes. Quelles théories faut-il choisir ? Peut-on penser à une adaptation dont les règles sont extraites automatiquement à partir des données ?

Göbel et al (Göbel et Mehm, 2013) ont proposé une adaptation qui consistait en la personnalisation du séquencement des scènes de jeu et des feedbacks pédagogiques par une génération automatique du contenu utilisant l'algorithme des n-grams (Andersen *et al.*, 2013) (sous-séquence de longueur n d'une suite plus grande). Luo et al (Luo *et al.*, 2014) ont proposé une modélisation de l'usager basée sur sa performance. Le modèle d'adaptation explicite visait à adapter automatiquement le contenu (génération automatique des scénarios) du système en utilisant les réseaux de neurones. Arnold et al (Arnold *et al.*, 2013) ont construit un modèle d'usager basé sur les styles d'apprentissage de Felder (Felder *et al.*, 1988). Les usagers étaient amenés à répondre au questionnaire de Felder, dont les réponses permettaient de déduire leur style d'apprentissage. Ils ont proposé un story-board comme modèle d'adaptation explicite basé sur une adaptation de contrôle. Cependant, bien que ce soit une solution prometteuse en matière de modèle d'adaptation, ce story-board est conçu manuellement puisqu'il implique l'intervention d'experts humains qui annotent chaque comportement pertinent observé chez les utilisateurs. Il ne serait donc pas d'une grande utilité pour des systèmes à grande échelle. Yannakakis et al (Yannakakis *et al.*, 2010) ont développé dans un jeux sérieux, une adaptation de contenu dont l'objectif était de générer automatiquement les artefacts du jeu en fonction du modèle de l'usager. Jerčić et al (Jerčić *et al.*, 2012) ont proposé une adaptation dont la difficulté des tâches était proportionnelle au niveau d'excitation qui est une dimension émotionnelle.

Plus l'usager est capable de contrôler et d'adapter son niveau d'excitation, plus l'environnement de décision devient facile. Vachiratamporn et al (Vachiratamporn *et al.*, 2014) ont conçu un système à base de règles comme moteur d'adaptation, visant à modifier le temps avant l'apparition de certains événements dans le système et leurs durées d'exécution. Mohammed et al (Al-Nakhal et Naser, 2017) ont également utilisé un moteur à base de règles pour concevoir un STI adaptatif permettant l'apprentissage des théories sur l'ordinateur. Dans le même sens, Hazem et al (Al Rekhawi et Abu Naser, 2018) ont utilisé un moteur de règles comme moteur d'adaptation dans un système visant l'apprentissage du développement des interface graphiques des applications Android. L'utilisation des règles de production (règles d'inférence) reste la technique la plus utilisée pour la conception des moteurs d'adaptation dans les systèmes interactifs en particulier ceux visant l'apprentissage. Ces règles sont en général, définies manuellement par les concepteurs. Un exemple de l'outil permettant la définition manuelle de règles de production est CTAT (Cognitive Tutor Authoring Tools) [2]. Il n'existe pas ou alors très peu de travaux dont ces règles de production ont été extraites automatiquement.

1.2 Approches de modélisation de l'usager

1.2.1 Modélisation de l'usager à partir des théories : explicite (stéréotypes)

Dans les approches guidées par les théories, la modélisation de l'usager est fondée sur des théories existantes. La modélisation explicite est surtout basée sur des taxonomies de stéréotypages des utilisateurs dont les plus populaires sont le *Big Five Personnality traits* pour la modélisation de la personnalité (Digman, 1990)(Ouverture, Conscienciosité, Extraversion, Agréabilité et Neuroticisme) et la taxonomie de Bartle (dans le contexte des jeux) (Bartle, 1996) sur les types de

2. http ://ctat.pact.cs.cmu.edu/

joueurs (4 types : les *tueurs* qui aiment provoquer des drames, les *compétitifs* qui sont compétitifs et aiment relever des défis difficiles, les *explorateurs* qui aiment explorer le monde et les *socialisateurs* qui sont souvent plus intéressés à avoir des relations avec les autres joueurs qu'à jouer le jeu lui-même). Ces typologies se basent sur les comportements des utilisateurs, leurs styles de jeux/apprentissage et leurs préférences. Généralement, les systèmes qui adoptent ces théories font passer un pré-test aux usagers pour déterminer au préalable à quelle catégorie ils appartiennent. Peu de travaux ont tenté de les extraire automatiquement à l'aide des techniques de fouille de données (classification bayésienne naïve (Keshtkar *et al.*, 2014), fouille de textes (Abyaa *et al.*, 2018; Mairesse *et al.*, 2007)) afin d'éviter à l'usager le remplissage d'un questionnaire qui peut vite devenir un obstacle à la motivation. Le BrainHex (Nacke *et al.*, 2014), qui est une amélioration de la taxonomie de Bartle a été proposé. Contrairement aux typologies précédentes, le BrainHex est générique et n'est donc pas lié à un type de système spécifique (Monterrat *et al.*, 2015). En outre, il a été conçu à partir d'un sondage en ligne qui a été effectué par plus de 60.000 utilisateurs. Le BrainHex établi sept différents types d'usagers des jeux : Seeker qui aime explorer les choses et découvrir son environnement, Survivor aui aime les scènes terrifiantes, Daredevil qui aime prendre les risques et sont excités par les frissons, Mastermind qui aime résoudre les puzzles et élaborer des stratégies, Conqueror qui aime vaincre des adversaires difficiles et surmonter les difficultés, Socializer qui aime interagir avec les autres, et Achiever qui est motivé à compléter la tâche et à collecter les choses. Comme dans les autres typologies, l'usager se voit attribuer un type précis à partir de ses réponses à un questionnaire. Outre les théories sur le stéréotypage, d'autres théories qui reviennent souvent dans la modélisation de l'usager portent sur la dimension affective. Parmi ces théories, nous pouvons citer le modèle émotionnel de Frome (Frome, 2007), les dimensions émotionnelles dont l'excitation et de la valence (Feldman, 1995), le modèle circonflexe de Russel (Russell, 2003) ou encore

le célèbre modèle d'Ortony-Clore-Collins (OCC model of emotions (Ortony *et al.*, 1988)), introduits plus tôt dans ce chapitre.

1.2.1.1 Modélisation de l'état cognitif

Puisque les systèmes interactif d'apprentissage humain ont la capacité de changer les processus cognitifs des usagers (Wouters *et al.*, 2013), plusieurs travaux se sont penchées sur la modélisation cognitive. Dans le domaine de l'AIED, la modélisation de l'état cognitif est basé entre autre sur les compétences, la performance dans le système intelligent et le niveau de connaissance de l'usager. Les travaux de modélisation et d'adaptation de Butler et al (Butler *et al.*, 2015) se sont largement inspirés des travaux du domaine de l'AIED. Ils ont conçu un modèle cognitif implicite de l'usager basé sur l'état de ses connaissances qui a servi d'input au modèle d'adaptation. Ce dernier était constitué des séquences de différentes actions effectuées par les usagers (*procedural traces*), le but étant de proposer à l'utilisateur un ajustement automatique du niveau de jeu ou d'une activité d'apprentissage ou même une activité quelconque s'il ne s'agit pas d'un STI, correspondant à son état de connaissances actuel. Wang et Al (Wang *et al.*, 2018) ont proposé une nouvelle famille de modèles d'apprentissage qui intègre un modèle de diagnostic cognitif et un modèle de Markov caché d'ordre supérieur pour la modélisation du processus d'apprentissage dans les systèmes d'apprentissage. Cette solution comprends des co-variables pour modéliser la transition des compétences dans l'environnement d'apprentissage. Göbel et al (Göbel et Mehm, 2013) ont proposé une modélisation de l'usager axée sur la dimension cognitive inspirée de la théorie KST (Knowledge Space Theory) (Albert et Lukas, 1999) visant à représenter l'état de connaissances de l'usager.

Dans le domaine des STIs, il existe plusieurs approches de représentation de l'état cognitif de l'apprenant dont le traçage de modèle et le traçage des connaissances.

Le traçage de modèle en anglais *model tracing*, est une technique utilisée pour le traçage de connaissances, basé sur le processus de construction de connaissances de l'usager en vue de lui donner une rétroaction spécifique (Anderson *et al.*, 1992). Dans ce modèle, les connaissances de l'expert sont représentées sous forme de règles de différents niveaux qui font état des divers niveaux de décision qu'un expert peut activer pendant l'accomplissement d'une tâche. Par cette approche, on associe chaque action de l'apprenant avec la règle de l'expert ou la règle qui peut le mieux représenter le type d'erreur commise. Cette approche de modélisation a été intégrée dans plusieurs STIs, notamment des systèmes d'apprentissage de la programmation en LISP (Reiser *et al.*, 1985). Certaines limites doivent cependant être soulignées, telles que le fait de laisser très peu de liberté à l'usager quant au contrôle de son cheminement d'apprentissage, la difficulté posée par les connaissances qui ne se représentent pas par des règles, le fait de ne pas permettre un apprentissage réellement basé sur l'erreur ou encore le manque de possibilités de réflexion sur l'action choisie (Giardina et Laurier, 1999). Le traçage des connaissances, considéré comme l'approche la plus populaire pour modéliser l'état actuel des connaissances de l'usager, vise à modéliser l'évolution des connaissances des apprenants pendant l'apprentissage (Corbett et Anderson, 1994). Les connaissances de l'apprenant sont modélisées comme une variable latente et sont mises à jour en fonction de ses performances sur des tâches données. Il peut être formalisé comme suit : étant donné les interactions $x_1, ..., x_t$ d'un apprenant sur une tâche d'apprentissage particulière, comment se comportera-t-il à $t+1$? Le but étant d'estimer la probabilité $p(x_{t+1}|x_1, ..., x_t)$ —probabilité de donner une réponse correcte sur x_{t+1} étant donné les réponses sur x_t ... x_1. Il existe de nombreuses solutions pour estimer cette probabilité, comme le *Bayesian Knowledge Tracing* (BKT) et le *Deep Knowledge Tracing* (DKT) qui sont présentés plus tard dans ce chapitre.

Dans le domaine des jeux, il y a également plusieurs facons de modéliser l'état

cognitif. Le BrainHex qui se démarque tout particulièrement par son caractère générique et englobant, est une taxonomie pouvant se classer dans la modélisation cognitive. Plusieurs systèmes interactifs en particulier les jeux sérieux, utilisent ces classifications prédéfinies pour modéliser le joueurs et adapter le jeu en conséquence. Nous pouvons citer les travaux de Bakkes et al (Bakkes *et al.*, 2012), qui se sont intéressés uniquement à la modélisation de l'usager conçu à partir de sa personnalité extraite du modèle du Big five, et de son style de jeu extraite du modèle de BrainHex. L'objectif était de prédire le plaisir, la motivation et l'effort de chaque joueur-apprenant. Bien que l'objectif ayant été atteint, l'étude n'a porté que sur un seul jeu, et donc pourrait déjà limiter son utilisation dans des jeux sérieux de contextes différents. Monterrat et al (Monterrat *et al.*, 2015) ont conçu un modèle du joueur basé à la fois sur le BrainHex et sur l'état des connaissances du joueur-apprenant. Les auteurs ont opté pour une adaptation de présentation qui consistait à sélectionner un ensemble d'objets du jeu à présenter en fonction du modèle de l'usager. Malheureusement dans leur proposition, la technique d'adaptation est au préalable définie avec l'aide des experts, en fonction du type d'usager par exemple, pour un utilisateur de type conquérant, il faut lui afficher le chronomètre. Il n'y a donc aucune adaptation dynamique ; de plus le modèle de l'usager n'est pas mis à jour au fil des interactions. De ce fait, puisque l'usager peut développer durant les interactions, un comportement différent de celui identifié à l'aide du questionnaire qui a été rempli en dehors du jeu, il peut s'avérer que l'adaptation préétablie soit peu, voire pas du tout efficace à long terme. Al-Abri et al (Al-Abri *et al.*, 2019) ont proposé pour un système e-learning un modèle utilisateur basé sur les styles d'apprentissage combinées avec le modèle de superposition (*overlay model*).

L'attribution d'un type à un usager passe par l'analyse de ses réponses à des questionnaires prédéfinis pour chaque théorie. Cette façon d'inciter l'utilisateur à

remplir des questionnaires durant une séance de jeu ou d'apprentissage contribue à diminuer sa motivation et son intérêt pour le jeu ou l'utilisation du système et donc diminuer ses chances de pouvoir apprendre quelque chose (Yannakakis et Hallam, 2009). Le modèle de l'usager est d'une part statique durant toute la session d'interaction et de plus, ces systèmes ne considèrent en général pas l'adaptation basée dans un contexte d'apprentissage. Malheureusement parce que chaque usager a une préférence, un style de jeu, un comportement particulier, une vitesse d'apprentissage différente et qui diffèrent d'un système à l'autre (Charles et Black, 2004), l'adaptation devrait être basée non pas sur un stéréotype prédéfini au départ qui ne reflète pas nécessairement sa vraie nature, mais sur une représentation plus fidèle et évolutive de son état actuel. L'utilisation de ces typologies n'est donc pas toujours suffisante pour caractériser précisément le joueur dans sa globalité et elles ne permettent pas de tenir compte de toutes les différences qui peuvent exister d'un joueur à un autre, ou d'un système à l'autre. Ainsi, construire un modèle usager basé sur un stéréotypage préétabli peut s'avérer utile mais le problème inhérent est l'ajustement de ces classes de joueurs prédéfinies au contexte par exemple le type Survivor (aime les scènes terrifiantes) n'est pas approprié dans un jeu de puzzle, qui peut ne pas avoir un ancrage dans l'ensemble des données à considérer. Une solution dans ce contexte serait d'utiliser le traçage de connaissance qui est une approche qui a su prouver son efficacité au fil des ans et est facile à mettre en oeuvre en notant que l'architecture l'implémentant soit bien spécifiée au préalable.

1.2.1.2 Modélisation de l'état affectif

La modélisation de l'état affectif est probablement la facette ayant été la plus développée dans les systèmes d'apprentissage adaptatifs. Il existe de nombreuses modalités dont le verbal et le textuel, pour la détection des états affectifs (Ouellet, 2016; Santos, 2016). Les recherches tendent le plus souvent vers des données

physiologiques (*biofeedback*) et des expressions faciales (Blom *et al.*, 2014) pour la modélisation de l'état émotionnel du joueur. La revue de littérature dans cette section sera axée sur les travaux ayant considérés les émotions comme modèle affectif. Selon (Vachiratamporn *et al.*, 2014), deux composants sont nécessaires pour le développement de systèmes affectifs : la reconnaissance des émotions et l'adaptation affective. Le système doit être capable de reconnaître en temps réel ou en différé, les émotions exprimées par les usagers. L'état émotionnel joue un grand rôle dans la capacité d'apprentissage d'un usager et sa manière d'interagir avec l'environnement d'apprentissage. Ainsi, les émotions négatives telles que la frustration, l'ennui ou la colère entraînent la baisse de motivation et de l'effort (Wouters *et al.*, 2013) tandis que celles positives comme la joie peuvent produire l'effet contraire. Il est donc important de considérer cette dimension dans la modélisation de l'usager afin de pouvoir prévenir ou de réguler les émotions qui pourraient avoir un impact négatif sur l'apprentissage.

Il existe plusieurs théories computationnelle des émotions dont les plus utilisés sont le modèle OCC, les émotions de base d'Ekman, et le *Learning Spiral Model* de Kort et al (Kort *et al.*, 2001). Cette dernière développe un modèle d'apprentissage à quatre quadrants dans lequel les émotions changent pendant que l'apprenant se déplace dans les quadrants ; il permet d'explorer l'évolution affective de l'usager au cours du processus d'apprentissage. Bien qu'il y ait plusieurs théories sur les émotions, il n'existe pas d'approche claire pour leur détection (Santos, 2016).

Dans le domaine des jeux, nous pouvons citer notamment les travaux de Yannakakis et al (Yannakakis *et al.*, 2010) qui ont proposé un jeu adaptatif de résolution de conflits où le modèle du joueur était constitué à la fois de la dimension affective, cognitive et culturelle. Pour la dimension affective, ils se sont orientés vers la proposition de Scherer (Scherer *et al.*, 1984) sur la théorie des émotions. Ils ont donc considéré les 5 classes d'états affectifs selon cette théorie : les émotions (co-

lère, joie, honte, etc.), les humeurs (irritable, gai, etc.), les préférences et attitudes (haineux, aimant, affectueux, etc.), les positions interpersonnelles (distant, froid, etc.) et enfin les dispositions affectives (nervosité, anxiété, etc.). Jatupaiboon et al (Jatupaiboon *et al.*, 2013) ont utilisé les signaux EEG en temps réel pour classer les émotions (joie et tristesse). Ils suggèrent l'utilisation des données physiologiques (GSR, ECG, EEG) contrairement aux données faciales et verbales car les rapports verbaux et expressions faciales peuvent être biaisés par les usagers. L'électroencéphalogramme (EEG) est l'enregistrement de l'activité électrique sur le scalp ; Il mesure les variations de tension résultant des flux ioniques dans les neurones du cerveau. Muñoz et al (Muñoz *et al.*, 2011) ont développé un modèle émotionnel du joueur-apprenant implémenté par 3 réseaux bayésiens dynamiques dont chacun représentait l'une des 3 émotions de Pekrun (Pekrun *et al.*, 2002).

En général, la détection des émotions peut se faire de plusieurs manières, soit par annotation manuelle des émotions, soit à partir des données physiologiques, soit à partir des données d'expressions faciales, etc. Nous constatons que la majorité des recherches tend vers la détection des émotions en utilisant des outils intrusifs (Casque EEG, Électrocardiographe, etc.). Ces outils, bien qu'efficaces, peuvent cependant nuire à l'immersion, à la motivation du joueur (Santos, 2016) et entraîner une baisse de concentration menant à l'abandon et donc un échec d'apprentissage. Leur usage limite aussi la portée de leur utilisation à grande échelle. Il serait intéressant d'explorer davantage des solutions moins intrusives telles que la reconnaissance des expressions faciales (Savran *et al.*, 2013), l'analyse de la parole (Devillers et Vidrascu, 2006) ou du texte (Ezhilarasi et Minu, 2012) écrit ou verbal produit par l'usager, l'analyse des traces d'interaction (D'mello *et al.*, 2008) pour inférer sur l'état affectif de l'utilisateur. Dans une telle perspective, l'utilisation de l'apprentissage machine est d'un intérêt certain, comme nous le voyons dans la section suivante. Par ailleurs, dans le domaine de l'AIED, la modé-

lisation de l'état affectif de l'usager a été une véritable préoccupation dès le début des années 2000 avec de nombreux travaux visant à construire un profil affectif de l'apprenant. Plusieurs techniques ont alors été employées pour la prédiction des émotions entre autres les réseaux bayésiens dynamiques (Conati et Maclaren, 2009), les techniques de classification de base sur les expressions faciales (Sidney *et al.*, 2005), la prédiction des émotions à partir du mouvement oculaire (Jaques *et al.*, 2014) etc.

1.2.2 Modélisation de l'usager par apprentissage machine : implicite

L'utilisation d'algorithmes d'apprentissage machine à des fins de modélisation de l'utilisateur a attiré beaucoup d'attention par le passé (Billsus et Pazzani, 2000). En général, la croissance du volume de données disponibles et l'amélioration des approches d'apprentissage machine sont le moteur de l'essor récent de la recherche dans ce domaine. Dans les approches guidées par les données, les chercheurs utilisent donc des techniques d'apprentissage machine et de fouille de données, pour développer des modèles pour prédire les intentions et les comportements des usagers et extraire des classes d'usagers ou des groupes de comportements à partir des données d'interactions collectées dans le système. Chaque usager se voit ainsi attribué une ou plusieurs classes de réactions et comportements en fonction de ses actions. Les algorithmes les plus utilisés dans ce contexte sont généralement les K-Moyennes (Drachen *et al.*, 2016; Zakrzewska, 2008), les réseaux de neurones (Demuth *et al.*, 2014; Piech *et al.*, 2015; Zhang *et al.*, 2017a) (pour la prédiction des comportements), les réseaux et classifieurs bayésiens naifs (Stern *et al.*, 1999; Pardos et Heffernan, 2010) et les SVM (Séparateurs à Vaste Marge) (Hearst *et al.*, 1998; Muyuan *et al.*, 2004).

Avec un modèle de l'usager guidé uniquement par la théorie, la prise en compte

des fonctions et des caractéristiques prédéfinies de l'usager empêche une analyse profonde des comportements pouvant émerger dans le système et augmente le risque d'ignorer des modèles importants dans les données (Drachen *et al.*, 2016). En résumé de ces approches basées sur les théories, l'utilisation de classes et de propriétés prédéfinies pour la modélisation de l'usager devrait au moins être vérifiée, validée ou agrémentée par une analyse non supervisée dont une approche guidée par les données. Les objectifs premiers de l'utilisation des données dans la modélisation de l'usager sont la classification, le regroupement (*clustering*) et la prédiction des comportements émergents dans le système. Les algorithmes de fouille de données supervisés (SVM, arbre de décision, etc.) sont utilisés pour la prédiction et la classification des comportements, tandis que les algorithmes de fouille de données non supervisés sont utilisés pour des fins de regroupement de comportements.

Ainsi, Missura et al (Missura et Gärtner, 2009) ont développé un modèle permettant d'adapter dynamiquement la difficulté d'un jeu en fonction des *clusters* extraits à partir des données de jeu. Ils ont utilisé l'algorithme des K-moyennes pour créer des *clusters* d'utilisateurs, chaque cluster représentant un certain type de joueurs. Le modèle d'adaptation mis en oeuvre ici était implicite et de contrôle : à chaque cluster correspond un certain niveau de difficulté. Ils ont utilisé l'algorithme SVM et un modèle de régression pour la mise au point d'un modèle de prédiction visant à prédire à partir du comportement de l'usager dont ses actions effectuées, ses décisions prises, le cluster auquel il pourrait appartenir. Une fois ce cluster déterminé, la difficulté était modifiée en conséquence au fil du jeu. Ha et al (Ha *et al.*, 2011) ont proposé un modèle du joueur représentant les buts qu'il cherche à atteindre. Pour ce faire, ils ont mis au point un cadre basé sur l'utilisation du champ aléatoire de Markov logique (modèle graphique probabiliste non dirigé avec des structures déterminées par des formules logiques du premier ordre)

pour la reconnaissance de buts. La reconnaissance d'objectifs (Min *et al.*, 2016) est la tâche d'inférer les buts visés par les utilisateurs à partir des séquences d'actions observées. Elle a été utilisée dans le système tutoriel d'apprentissage *Crystal island* (Rowe *et al.*, 2009) où les buts des apprenants étaient inférés à travers leurs réponses aux questions que posait le système de façon narrative. Ce cadre a également été utilisé dans les jeux afin d'ajuster dynamiquement leurs comportements par rapport aux objectifs changeants des joueurs. Yannakakis et al (Yannakakis et Hallam, 2009) ont proposé un modèle d'adaptation en temps réel dont l'objectif était d'optimiser la satisfaction de l'usager. Un réseau de neurones artificiel et un algorithme de sélection de paramètres (*features selection*) ont été appliqués aux données provenant d'une enquête (réponses à des questionnaires administrés avant et pendant l'utilisation du système) et du journal d'activités (score, temps moyen de réponse, etc.), pour générer un modèle de prédiction de la satisfaction de l'usager. Shaker et al (Shaker *et al.*, 2010) ont proposé un modèle de génération automatique des niveaux du jeu Super Mario Bros, utilisant un réseau de neurones et en se basant sur un modèle d'expérience du joueur déduit à partir des données d'interactions. Drachen et al (Drachen *et al.*, 2012) ont proposé un groupement de différents comportements observables dans le système. Le but était d'obtenir des classes de comportements permettant de modéliser les usagers. Ils ont mis à contribution les algorithmes de fouille de données dont les K-Moyennes et le *Simplex Volume Maximisation* (SIVM). De même, Gow et al (Gow *et al.*, 2012) ont proposé une modélisation du joueur (représentant son style de jeu) construite à partir d'une approche d'apprentissage semi-automatique et non supervisée combinée à l'analyse linéaire discriminative multi-classes (LDA : multi-class Linear Discriminant Analysis) (McLachlan, 2004) appliqué aux *logs*.

L'émergence du *big data* et des techniques de machine learning ont ouvert plusieurs possibilités pour une modélisation plus précise des usagers. Ainsi les techniques

bayésiennes et les techniques neuronales se sont bien prêtées à la situation. Elles ont entre autres permis des résultats notables dans le domaine de la modélisation de comportements notamment le traçage de connaissances. Dans cette approche, la maîtrise d'une connaissance par un usager est traitée comme un état latent et la probabilité que ses connaissances se trouvent dans cet état est mise à jour en fonction du comportement observé (Halpern *et al.*, 2018). Ainsi, pour ce qui est des habiletés cognitives et du traçage de connaissances, Conati et al ont utilisé les algorithmes de classification pour prédire les habiletés cognitives des utilisateurs (Conati *et al.*, 2017). D'autres comme auteurs ont utilisé les réseaux bayésiens pour la modélisation du processus d'acquisition des connaissances (Tato *et al.*, 2017a; Zhang et Yao, 2018; Kantharaju *et al.*, 2018). Plusieurs autres auteurs se sont récemment concentrés sur le traçage de connaissances en utilisant les architectures d'apprentissage profond notamment les réseaux de neurones récurrents (RNN) et les mémoires à long et à court terme (LSTM) (Piech *et al.*, 2015; Wang *et al.*, 2017; Zhang *et al.*, 2017b; Montero *et al.*, 2018). Il est à noter que l'utilisation des réseaux bayésiens peut être vue comme une approche hybride puisque les réseaux peuvent être conçus manuellement par les experts du domaine (Conati *et al.*, 1997; Horvitz *et al.*, 1998; Rowe et Lester, 2010) ou conçus (architecture et tables de probabilités) automatiquement à partir des données (Friedman *et al.*, 1999; Tsamardinos *et al.*, 2006) en utilisant des approches d'apprentissage automatique.

Même si les méthodes de construction du modèle de l'usager guidées par les données sont très prometteuses car elles permettent d'extraire automatiquement des caractéristiques pertinentes, elles peuvent aussi extraire des informations dénuées de sens (Demuth *et al.*, 2014). Par exemple, ces méthodes peuvent extraire des relations non pertinentes telles qu'une relation entre l'âge et le choix d'un personnage dans un jeu, puisque ces deux informations auront tendance à se répéter

si plusieurs joueurs d'un même âge choisissent souvent un même personnage. La classification, la prédiction et le clustering des comportements des joueurs dans les jeux actuels deviennent de plus en plus ardus (Drachen *et al.*, 2016) en raison du volume, de la grande dimensionnalité et la multi-modalité des données de jeu. Également, les techniques fondamentales servant à l'apprentissage dans les réseaux de neurones ne sont pas forcément adaptées pour la modélisation des usagers et ne tiennent pas compte des connaissances a priori et a posteriori. De ce fait, il est nécessaire de (1) développer des solutions fondamentales adaptées au problème de modélisation de l'usager, et de (2) miser sur des techniques hybrides capables de tirer le meilleur des approches pilotées par les données (*data-driven* et des approches pilotées par les théories *theory-driven*).

Dans la modélisation explicite, il y a également certains travaux qui se sont penchés sur des solutions utilisant des techniques de planification dont la reconnaissance de plans. La reconnaissance de plans est basée sur le suivi de la performance des usagers en fonction de séquences d'actions des apprenants (tracage de modèle). Les actions de l'usager sont adaptées à une bibliothèque de tous les plans possibles. Le plan le plus proche des actions de l'utilisateur sera choisi comme modèle de l'usager. Nous n'allons pas présenter les travaux s'alignant vers cette solution car dans cette approche, il est très coûteux de créer une bibliothèque de plans et cela nécessite des calculs complexes et un stockage de données important. La mise en oeuvre de cette méthode nécessite un algorithme complexe (Tadlaoui *et al.*, 2016).

1.3 Analyses critiques et pistes de solutions

1.3.1 Vers une modélisation riche de l'usager ?

D'une manière globale, nous constatons que les travaux de recherche sur la modélisation de l'usager en particulier du joueur-apprenant pour des fins d'adaptation dans les systèmes visant l'apprentissage ne prennent généralement en compte que des profils d'usagers de très hauts niveaux. Des études antérieures ont porté sur les types d'usagers en termes de stéréotypes taxonomiques des comportements en considérant différents styles de jeu, d'apprentissage et préférences (Bakkes *et al.*, 2012; Alexander *et al.*, 2013; Tondello *et al.*, 2016; Truong, 2016). Certaines études ont proposé une vue synthétique des stéréotypes existants en terme de méta-typologie pour une meilleure utilisation (Hamari et Tuunanen, 2014; Nacke *et al.*, 2014). Un profil est assigné à un usager donné en se basant sur son comportement capturé parfois à l'extérieur ou à l'intérieur du système (Yannakakis et Hallam, 2009; Yannakakis *et al.*, 2013). Bien que le BrainHex ait été utilisé avec succès dans certaines études, il n'est pas aussi stable, et théoriquement fondé comparativement à la théorie du Big Five. Un des avantages des approches stéréotypées est que les profils se concentrent sur une cible et ont donc tendance à être simples et faciles à gérer dans l'adaptation puisque à chaque stéréotype va correspondre un certain nombre de règles d'adaptation. Cependant, il existe plusieurs inconvénients à utiliser ces approches, dont les suivants : 1) Les profils peuvent forcer la combinaison de plusieurs styles d'apprentissage différents et ainsi, parfois causer la perte de distinctions individuelles importantes ; 2) Il est possible que l'usager se comporte différemment dans un environnement par rapport à la réalité, et donc, les approches où les caractéristiques ont été fixées en dehors du contexte (ex : la théorie de Bartle) pourraient ne pas convenir dans un contexte différent ; 3) Les caractéristiques importantes de l'apprentissage (ex : les compé-

tences et les théories cognitives / d'apprentissage sous-jacentes) sont généralement ignorées dans ces approches, ce qui les rend parfois inadaptés pour les systèmes d'apprentissage.

Dans le domaine de l'AIED, il est maintenant largement reconnu que des modèles d'utilisateurs plus sophistiqués sont nécessaires pour fournir une assistance plus souple et adaptative qui répond aux besoins individuels des apprenants. De plus, le besoin d'une approche holistique de la modélisation des usagers a récemment été souligné dans certaines recherches, comme un point-clé pour la conception d'un modèle d'adaptation efficace au maximum (Bontchev, 2016; Musto *et al.*, 2018; Michel et Matthes, 2018). En conséquence, considérer un modèle de l'usager enrichi est une avenue prometteuse pour la recherche sur une adaptation de haute qualité. Dans ce sens, Yannakakis et al (Yannakakis *et al.*, 2013) définissent la modélisation du joueur-apprenant comme un processus impliquant toutes les dimensions permettant de le caractériser entre autres dimensions cognitives, affectives et sociales. Il est important d'identifier les caractéristiques clés associées à chacune de ces dimensions, qui sont nécessaires pour garantir l'efficacité de l'adaptation en terme d'expérience et en terme de gain d'apprentissage. L'approche *top-down* de l'élicitation des caractéristiques devrait être régie par des principes appropriés de théories pertinentes telles que la théorie de l'OCC sur l'émotion pour classer les états affectifs de l'usager. Cependant, les caractéristiques induites de la plupart des théories peuvent ne pas avoir un ancrage pertinent dans les données et dans le contexte. L'approche *bottom-up* vise à extraire des caractéristiques pertinentes dans les données recueillies pendant les sessions de jeu en utilisant les algorithmes de fouille de données et d'apprentissage machine. Une telle approche est plus objective et potentiellement plus précise, mais seulement à condition de recueillir suffisamment de données. En outre, cette approche peut susciter des problèmes d'interprétation des caractéristiques extraites puisque les techniques de fouille de

données sont en général des boîte noires puisque les résultats ne sont pas souvent humainement interprétables. Bien qu'il soit fréquent que les modèles des usagers développés soient utilisés pour assigner un stéréotype prédéfini, nous avons l'intention de traiter chaque joueur indépendamment afin d'optimiser l'environnement pour lui. Les objectifs premiers d'un système d'apprentissage est l'amélioration de l'apprentissage. De plus nous constatons une montée croissante de conception de systèmes adaptatifs axés sur un modèle de «*Design by intuition*» ce qui freine énormément l'éclosion du domaine et son application à grande échelle. Bien qu'il existe différentes approches pour modéliser les usagers, on en sait peu sur les caractéristiques les plus informatives pour mieux les représenter, afin de guider les décisions de conception et d'adaptation en conséquence (Poeller *et al.*, 2018). Dans cette thèse, nous proposons des algorithmes d'apprentissage machine efficaces et adaptés à la modélisation des usagers et qui de plus combinent données brutes et connaissances liées aux théories. Nous visons des solutions réutilisables dans plusieurs contextes puisque plusieurs dimensions qui caractérisent un usager ne changent pas d'un système à un autre.

1.3.2 Vers un modèle d'adaptation dynamique

Même avec un modèle précis et holistique de l'usager, il est toujours important de se pencher sur les méthodes d'exploitation de la richesse de ce modèle pour mieux adapter le système. Bien qu'il n'existe pas de méthodes et d'outils standards pour une adaptation efficace, il y a déjà des progrès significatifs dans la tentative de mieux adapter le système aux besoins des usagers. Ces besoins peuvent être inférés à partir des caractéristiques fixes telles que la personnalité et la dimension sociale, et des variables affectives, cognitives et métacognitives. Selon le type de système, l'adaptation peut être implémentée implicitement ou explicitement. De plus, elle peut se limiter à la présentation : affichage d'objets spécifiques (Monterrat *et al.*,

2015) ; au contrôle : la progression du jeu (Missura et Gärtner, 2009), ajustement dynamique de la difficulté (Butler *et al.*, 2015), les comportements des personnages non joueurs (NPCs) (Lim *et al.*, 2012) ; ou au contenu : la génération de niveaux de jeu personnalisés (Hendrikx *et al.*, 2013). Suivant la perspective holistique du modèle de l'usager, nous considérons l'adaptation comme un processus actif qui fonctionne sur des informations dynamiques obtenues sur l'usager. Par conséquent, au lieu d'essayer de créer un mécanisme universel permettant à un système de s'adapter à un type d'usager particulier, nous allons développer une solution hybride dont les règles d'adaptation proviennent à la fois des experts et des données.

Puisque les STI ont fait grandement progresser le potentiel d'enseignement par ordinateurs grâce à des techniques d'IA qui permettent de suivre l'évolution des connaissances des apprenants et de choisir les problèmes d'apprentissage les plus appropriés (Butler *et al.*, 2015), nous nous en inspirons pour le développement de notre modèle usager et de notre modèle d'adaptation.

1.4 Conclusion

Après cette revue de littérature, nous constatons que de nombreux défis persistent pour la modélisation de l'usager dans les systèmes interactifs, en particulier ceux visant l'apprentissage dont les STIs et jeux sérieux, et l'adaptation. Les solutions existantes sont pour la plupart (1) des solutions propriétaires ; (2) ne tirent pas profit des avancées dans le domaine de l'apprentissage machine (en particulier l'apprentissage profond) ; (3) ignorent les travaux qui ont été abattus pendant des années de recherches (connaissances a priori et posteriori) (4) ne considèrent pas la multidimensionnalité du comportement humain. Ce chapitre nous a également permis de comprendre le besoin d'aller vers des systèmes hautement adaptatifs

qui tirent profit d'un modèle riche de l'usager.

Considérer toutes les facettes de l'usager dans sa modélisation implique d'avoir à traiter des données de plusieurs modalités. Nous devons à cet effet utiliser voire développer de nouveaux algorithmes d'apprentissage machine et de fouille de données, nécessaires à l'extraction de caractéristiques pertinentes et à la modélisation. Le modèle qui en découle doit ainsi être à mesure de représenter le joueur dans le système avec lequel il interagit et ce en étant capable de prédire ses comportements, actions futures, etc. D'autre part, la mise au point d'un modèle d'adaptation dynamique implique également le développement d'algorithmes capables de prendre en considération le modèle de l'usager et de proposer des actions à prendre dans le système pour y répondre de façon à le garder immergé, motivé et à assurer un gain d'apprentissage positif. Les chapitres suivants sont axés sur les solutions que nous proposons. Nous présentons en détail notre méthodologie, nos propositions ainsi que des applications concrètes de nos solutions pour des fins de validation.

CHAPITRE II

MÉTHODOLOGIE ET CADRE EXPÉRIMENTAL

Ce chapitre a pour but de présenter le cadre méthodologique et expérimental de notre recherche. Tout d'abord, nous présentons la méthodologie de recherche axée sur le développement et l'évaluation de la performance des solutions (conçues), dans l'intention explicite d'améliorer le rendement fonctionnel de la solution. Par la suite, nous présentons chacune des 4 itérations de notre projet, les résultats attendus et l'approche ou le protocole utilisé pour les obtenir. Les résultats finaux des itérations sont brièvement résumés dans le présent chapitre, mais sont présentés plus en détail dans les 3 prochains chapitres.

2.1 Introduction

Notre démarche méthodologique est inspirée de la *design science research methodology (DSRM)* (Peffers *et al.*, 2007). Le DSRM est une méthodologie adaptée pour les projets de recherche dans le domaine de l'algorithmique et de l'interaction homme-machine. Le processus du DSRM comprend six étapes : identification et motivation du problème (Introduction), définition des objectifs d'une solution (Introduction), conception et développement (chapitre 3,4), démonstration, évaluation (chapitre 5) et communication (chapitre 6).

Nous rappelons que cette thèse vise à étendre les recherches antérieures sur l'apprentissage machine pour la modélisation des usagers et l'amélioration de l'adaptation dans les systèmes interactifs visant l'apprentissage humain. Afin de concevoir le modèle de l'usager escompté, nous devons recueillir un ensemble de données d'interactions nous permettant d'extraire les caractéristiques importantes qui caractérisent le joueur-apprenant. Certaines caractéristiques seront définies manuellement par les experts et d'autres seront extraites à partir des données. Ces caractéristiques sont associées aux différentes dimensions considérées dans le modèle. Par ailleurs, nous aurons besoin d'une part des données d'interactions et d'autre part d'une plate-forme interactive d'apprentissage fonctionnelle dans laquelle nous pourrons tester et raffiner nos solutions. Pour cela, nous développerons un jeu vidéo sérieux approprié, offrant un contexte riche d'un point de vue émotionnel, social et cognitif. Également, nous utiliserons un autre système tutoriel existant visant à l'apprentissage du raisonnement logique (Nyamen Tato, 2016). Le processus méthodologique général de notre recherche est présenté à la figure 2.1 que nous détaillons dans les prochaines sections.

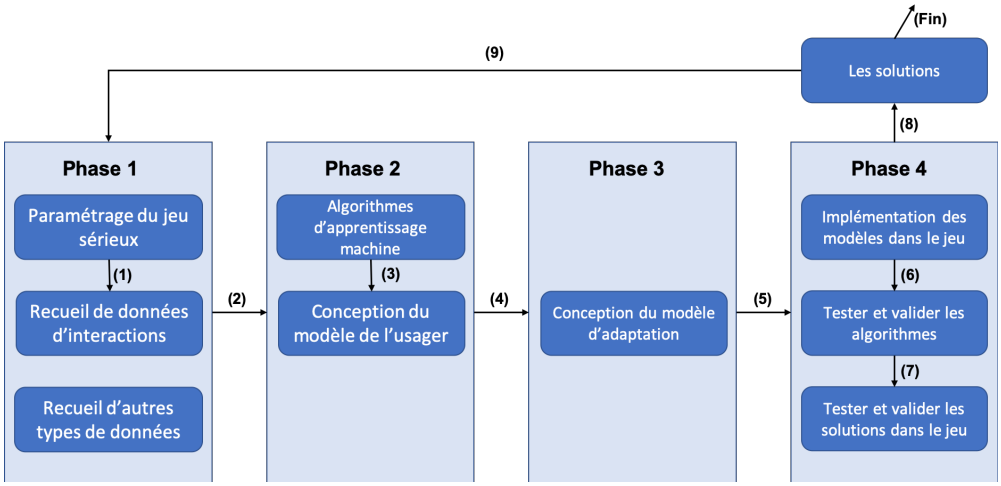


Figure 2.1 Processus méthodologique du projet de recherche.

Dès le départ, nous disposons de 2 environnements virtuels interactifs d'apprentissage : (1) le jeu sérieux *LesDilemmes* dont l'objectif est d'évaluer et d'améliorer le raisonnement socio-moral du joueur ; (2) le système tutoriel intelligent Muse-logique (Nyamen Tato, 2016) dont l'objectif est d'évaluer et améliorer les compétences en raisonnement logique de l'apprenant. La phase 1 de notre travail à consister à d'abord paramétrer le jeu, c'est-à-dire permettre la capture à travers le jeu des différentes réactions et actions que nous voulons observer (réactions émotives, comportement social, cognitif et métacognitif). En second lieu, nous avons conduit une première expérimentation avec le jeu (version non adaptative) dans le but de recueillir les données multimodales utiles au développement de nos modèles. Par la même occasion, nous avons recueillis (pour des fins de validation et de test), d'autres données disponibles dont celles provenant de Muse-logique et de bases de données publiques. La phase 2 visait à concevoir les algorithmes et architectures utiles pour la conception du modèle de l'usager. Ceci a été fait à partir des théories existantes (en lien avec les différentes dimensions que nous avons considéré pour la modélisation de l'apprenant-joueur) et à partir des techniques d'apprentissage machine et de fouille de données multimodales. La phase 3 visait le développement d'un modèle d'adaptation dynamique capable à partir des états comportementaux de l'apprenant-joueur, de proposer des solutions d'adaptation personnalisées (en réponse aux états prédits). La phase 4 visait à la validation des modèles proposés. Il a s'agit dans cette phase de tester la performance des algorithmes proposés d'abord en tant que solutions individuelles ensuite en tant que solutions intégrées dans un système interactif d'apprentissage. Cette dernière validation à consister à tester l'efficacité du mécanisme d'adaptation dans la version ajustée du jeu. Nous avons prévu une boucle itérative dans notre processus afin de raffiner davantage les solutions proposées.

2.2 Phases de la recherche

2.2.1 Phase 1 : Paramétrage du jeu sérieux et recueil de données

Nous avons développé un jeu sérieux appelé *LesDilemmes* dont l'objectif est d'évaluer et d'améliorer les compétences en raisonnement socio-moral du joueur-apprenant. Ce jeu est un prolongement des travaux de Miriam Beauchamp qui a travaillé au développement d'un test sur le développement du raisonnement socio-moral (Le test SoMoral, (Dooley *et al.*, 2010)), où l'individu (très souvent un adolescent) est amené à décider quelle action il prendrait face à un dilemme moral et à justifier verbalement son choix. Les experts utilisent alors les raisons évoquées par le joueur pour émettre un jugement sur son degré de maturité. Le degré de maturité (qui va du stade 1 au stade 5) est déterminé en fonction du contenu du justificatif de l'individu et des informations extraites du système de cotation construit par des experts psychologues (Chiasson *et al.*, 2017). Un extrait de ce système de cotation est présenté à la figure 2.2. Cette mesure est faite manuellement pour des fins d'évaluation. L'un de nos objectifs était de l'automatiser puisque c'est l'élément de connaissance à apprendre dans le jeu et elle doit pouvoir être tracée en temps réel pour la mise à jour du modèle de l'apprenant joueur. Chacun de ces stades corresponds à :

- Score 1 : Orientation vers la punition et l'obéissance à l'autorité
- Score 2 : Orientation vers les échanges égocentriques
- Score 3 : Orientation vers les relations interpersonnelles
- Score 4 : Régulation de la société (justifier ses actions sur la base de normes)
- Score 5 : Évaluation du contrat social (justifier sur la base d'évaluation critiques)
- Score 0 : Réponse non évaluable

Il peut arriver que l'expert humain n'arrive pas à attribuer un score à un verbatim

Level	Brief description	Example
1	Moral justifications have an egocentric focus, which is based on obedience to higher authorities and potential consequences to themselves for their actions (e.g. punishment). Thinking at this level is inflexible; there is only one right/wrong way to act	• Because I could go to jail
2	Moral justifications are based on a concept of pragmatic deals or exchanging favors with others ('fair deals'). Thinking is more flexible and is determined by context. The correct option is the one that is right for oneself (self-interest)	• Because I might need his/her help in the future
3	Moral justifications have a focus on interpersonal relationships, a sense of 'good-ness', and feelings such as empathy and trust. Decisions are made with good motives and a prosocial perspective of the world	• Because he/she could get hurt
4	Moral justifications start to incorporate a broader view of morality; based on the compliance with rules, regulations and standards that society has established to ensure social order	• Because if everyone were to be unfaithful, relationships would not have any meaning
5	Moral justifications are characterized by the capacity to evaluate situations from various points of view to identify values involved in the specific situation to make the fairest decision. Protection of fundamental values and people's rights is specific to this stage, even though these concepts are expressed very concretely	• Because people work hard for their things and we should respect their belongings

Figure 2.2 Brève description des niveaux de raisonnement (codage So-Moral) avec quelques exemples de descriptions (Chiasson *et al.*, 2017).

car il n'y a pas assez d'informations pertinentes pour lui permettre de classer le verbatim dans l'un des 5 stades. À cet effet, il attribue un score de 0. Également, il peut arriver qu'un verbatim se voit attribué un stade intermédiaire. Ainsi les experts dans ces cas mettent 1.5, 2.5, 3.5 ou 4.5 selon le verbatim. Cette spécification est accompagnée d'une liste de concepts clés pour chaque stade ainsi que d'un descriptif textuel d'une page pour chacun des stades. Le jeu a été implémenté sous forme d'environnement 3D, l'objectif étant de recréer le plus fidèlement possible la situation réelle dans laquelle l'individu aurait normalement à faire ses choix dans un environnement réaliste avec des amis qui l'entourent et d'autres personnes, entre autres figures d'autorité, comme le professeur ou le parent. Ainsi dans le jeu, le personnage joueur est entouré de personnages non-joueurs qui représentent différents stades de maturité moral et qui présentent chacun les justifications qui sont extraites des verbalisations recueillies précédemment et classées selon la grille du *SoMoral*. Le jeu demande au joueur de décider ce qu'il ferait, de dire verbalement pourquoi (ce qui est enregistré) mais aussi de consulter les PNJs et d'évaluer les justifications qu'ils suggèrent. Le paramétrage de cet environnement dans la première phase à consister simplement à mettre en place le système d'évalua-

tion de l'apprentissage et la connexion avec les différentes sources de données : le *Facereader* pour les émotions et un microphone pour les données verbales. Les données multimodales sur lesquelles nous avons travaillé pour ce cadre applicatif proviennent donc de ces sources mais également des interactions qui sont enregistrées durant le jeu (les évaluations des PNJ, le score etc.). Une autre partie de cette phase 1 était concentré au recueil d'autres données utiles, entre autres celles provenant de Muse-logique et celles produites par l'équipe du laboratoire ABCs (Aptitudes cognitives Bilan Cerveau Socialisation) à l'Université de Montréal qui travaille sur le SoMoral.

2.2.2 Phase 2 : Algorithmes et architectures pour la conception du modèle de l'usager

Nous avons fait l'hypothèse qu'un modèle riche de l'usager doit être constitué à la fois d'une dimension affective, cognitive et sociale. Comme nous l'avons mentionné dans la revue de littérature, la conception de ce modèle est fait par une approche hybride. Ainsi, notre flux de travail général pour cette phase 2 (voir figure 2.3) à consister dans un premier temps en la sélection à partir des théories, des différents attributs qui permettent de caractériser les états affectifs, cognitifs et sociaux de l'usager. En second lieu, d'autres propriétés comme les caractéristiques latentes non interprétables sont extraites directement à partir des données du jeu à l'aide d'architecture d'apprentissage automatique. Ces architectures qui permettent la prédiction des caractéristiques pertinentes, servent pour la conception du modèle de l'usager. Ici, nous avons développé tout d'abord un nouvel algorithme d'apprentissage profond (solution d'optimisation) nécessaire à l'adaptation de ce type de solutions dans un contexte de modélisation du comportement humain. Ensuite nous avons utilisé cet algorithme pour développer nos différentes architectures neuronales pour la modélisation cognitive, affective et sociale de l'usager.

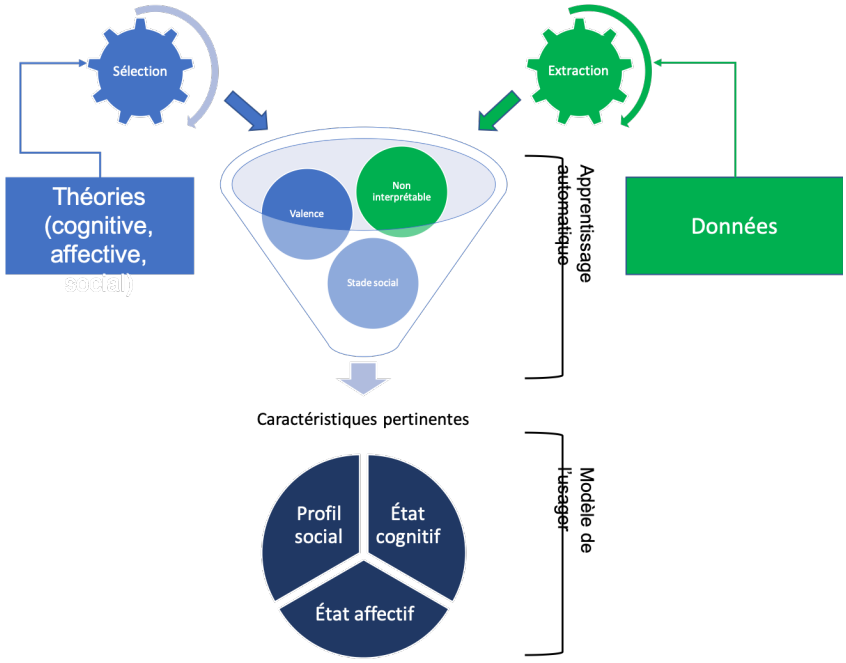


Figure 2.3 Flux général pour le développement d'un modèle usager riche.

2.2.2.1 Sélection et extraction des attributs de l'usager

Nous entendons par attributs, propriétés ou caractéristiques de l'usager, toutes les informations capables de le décrire lui-même (ex : son âge, son sexe) son comportement (ex : méthode d'apprentissage, décisions), ses émotions (ex : valence, niveau d'excitation), ses habiletés cognitives.

Approche guidée par les théories (sélection manuelle d'attributs) : Tout d'abord, l'hypothèse du modèle riche de l'usager est affinée en explorant ce que nous disent les théories par rapport aux caractéristiques associées à chacune des facettes visées (cognitive, sociale et affective). La dimension affective est l'une des dimensions les plus importantes de notre modèle de l'usager puisque les émotions permettent de motiver davantage, d'accroître l'apprentissage et la compréhension des joueurs apprenants (Missura et Gärtner, 2009). Pour la reconnaissance des

états affectifs et sur la base des travaux antérieurs en informatique affective, nous nous sommes limités à l'utilisation de la reconnaissance faciale. La théorie des émotions que nous utilisons est celle du modèle circonflexe des émotions selon Russel (Posner *et al.*, 2005), tout d'abord parce qu'elle est simple à mettre en place, mais également parce qu'elle a été validée et utilisée avec succès par plusieurs autres chercheurs dans le domaine des jeux sérieux. Dans cette théorie, les émotions sont classées en leurs composants indépendants qui sont l'excitation et la valence. Nous n'avons pas à recalculer ces caractéristiques puisque l'outil sélectionné nous les fournies.

La dimension cognitive est régie par la théorie ACT-R (Atomic Component of Thought – Rational) d'Anderson (Anderson *et al.*, 1997) qui a été largement utilisé comme base pour les modélisations cognitives et pour la conception de tuteurs cognitifs dans le domaine de l'AIED et tout récemment dans les environnements de jeu (Kennedy et Krueger, 2013; Smart *et al.*, 2016). Elle possède des caractéristiques fondées sur la littérature en recherche expérimentale et qui font d'elle une théorie incontournable dans la modélisation de la performance cognitive humaine (Smart *et al.*, 2016). C'est une architecture cognitive symbolique à base de règles, capable de simuler/reproduire le fonctionnement cognitif. La connaissance y est codée dans deux mémoires différentes : une mémoire procédurale, exprimée sous forme de règles de production, et une mémoire déclarative, exprimée sous la forme d'un réseau de nœuds liés. L'état de connaissance de l'usager étant sujet à changement, la compétence doit être assignée avec un certain degré de certitude ; ainsi le modèle cognitif de l'usager ne peut être qu'une approximation de son modèle réel. Il est donc important que le diagnostic soit soutenu par un formalisme qui permette des déductions certaines à partir d'incertitudes (Mayo et Sheppard, 2001). Les réseaux de neurones séquentiels (RNN et LSTM) sont adéquats pour la tâche : ils permettent d'inférer la probabilité de maîtriser une compétence à partir d'un

modèle de réponse spécifique (Piech *et al.*, 2015). Nous avons construit donc des réseaux de neurones pour la modélisation de l'état cognitif du joueur-apprenant.

L'intégration d'une dimension sociale dans les jeux sérieux est importante, les interactions sociales étant devenues incontournables dans la plupart des jeux, mais aussi dans le cadre de l'apprentissage. Le profil social de l'usager que nous mettons en place vise à intégrer cette dimension dans les jeux sérieux et donc de permettre une adaptation sociale. Très peu d'auteurs s'intéressent au profil social de l'usager si ce n'est à travers sa personnalité. Or, comme nous l'avons si bien remarqué dans le chapitre précédent, les jeux qui tiennent compte de la personnalité de l'usager, obligent ce dernier à remplir des questionnaires avant ou pendant la séance de jeu, ce qui est un frein majeur à l'immersion, la motivation et donc l'apprentissage. Notons que la dimension personnalité / profil social du modèle rapporte des informations statiques qui ne sont pas directement liées au jeu. Toutefois, puisque la personnalité peut influencer les émotions (Saklofske *et al.*, 2012) et donc le comportement de l'utilisateur dans un jeu (ses actions, sa croyance, son attitude), elle est considérée comme un paramètre à intégrer.

En plus de la personnalité, nous introduisons une nouvelle métrique de mesure du profil social, qui est basée sur la théorie du So-Moral (Beauchamp *et al.*, 2013) et qui fournit des principes pour déterminer les compétences en raisonnement socio-moral chez l'humain. Le raisonnement socio-moral est l'une des fonctions cognitives essentielles pour la prise de décision ainsi que pour l'adaptation sociale. Il s'agit d'une dimension importante du comportement humain puisqu'elle contribue au développement de ce dernier (Dooley *et al.*, 2010). Étant donné que le jeu sur lequel nous travaillerons vise à développer chez les joueurs cette compétence-là, elle devra donc être intégrée. Nous pensons que si le jeu s'adapte en fonction du niveau de maturité sociale d'un joueur-apprenant, alors le gain d'apprentissage serait encore meilleur. Par exemple, pour un joueur ayant un raisonnement moral

orienté vers la punition et l'obéissance à l'autorité (Niveau 1 dans le So-Moral), on pourrait envisager d'intégrer dans le scénario de jeu un PNJ d'autorité. Dans le So-Moral, le niveau de maturité d'un individu est déterminé à partir de ses justifications verbales émises lors de résolutions de dilemmes moraux. Il s'agit alors ici d'implémenter un outil de mesure automatique de ce niveau dans les jeux. Les solutions existantes pour la fouille de textes supervisée (SVM, *Latent Semantic Analysis*, *Naive Bayes* et *Latent Dirichlet Allocation*) ont été sans succès (*accuracy* variant de 19 à 55 % et une F-mesure variant de 0.005 à 0.15) sur nos données. Il y a donc nécessité à développer un algorithme original de fouille de textes supervisée tenant compte du contexte (concepts clés décrivant chaque stade) et d'un ensemble de données déjà étiquetées puisque nous avons à notre disposition un corpus de plus de 1000 verbatims annotés par les experts.

Approche guidée par les données : Certaines théories nous guident sur les caractéristiques à observer chez l'usager. Ces attributs que nous élicitons 'manuellement' sont importants pour la caractérisation de l'usager car ils fournissent des informations sémantiques de haut niveau. Cependant, l'élicitation manuelle des attributs est subjective et les attributs potentiellement utiles (discriminatifs) peuvent être ignorés. D'où le besoin d'utiliser les données de jeu pour extraire des caractéristiques objectives et pertinentes pour la prédiction du comportement. Dans ce dernier cas, on parlera plutôt de caractéristiques latentes ou cachées car elles ne sont pas directement observables dans les données. Les attributs ayant un sens sémantiquement parlant pour les humains ne correspondent pas nécessairement à l'espace des attributs détectables et discriminatifs (Fu *et al.*, 2013). Ainsi, dans cette étape, nous avons mené une étude visant à apprendre automatiquement de nouvelles caractéristiques qui pourraient être intéressantes dans la modélisation et pour l'adaptation. L'objectif étant de prendre en compte les aspects de l'usager et les spécificités des données du jeu non modélisées par les approches

théoriques sélectionnées. L'information dans le monde réel provient de plusieurs canaux d'entrée par exemple, les images sont associées à des légendes, les vidéos contiennent des signaux visuels et audio etc. (Srivastava et Salakhutdinov, 2012). L'information étant donc de nature multimodale, il convient d'en tenir compte pour en tirer le maximum de bénéfice. C'est dans ce même ordre d'idée que nous soutenons la thèse selon laquelle, pour avoir des informations pertinentes sur un usager, il est important de considérer toutes sources de données nous informant sur son état actuel. Les données que nous manipulons proviennent de l'expression faciale, de l'audio et du journal d'activités. Étant donné que nous faisons face à une situation où les données sont multi-sources (ou multi-modales), nous avons développer des techniques adaptées à ce contexte.

À cet effet, nous optons pour une approche *feature-level* dont le but est de combiner les caractéristiques extraites de chaque canal d'entrée en un «vecteur joint» (Poria *et al.*, 2015). Nous avons développer un modèle d'apprentissage machine capable d'extraire et fusionner des attributs de différentes modalités dans des données de jeux, en prenant en compte le fait que certaines modalités ou données peuvent être manquantes afin de faire des prédictions sur l'usager. Ce modèle est conçu à l'aide des approches du domaine de l'apprentissage profond (*Deep learning*). Ainsi, nous exploitons la capacité d'abstraction des architectures de ce domaine pour extraire des propriétés latentes (Yosinski *et al.*, 2014). Ce que nous proposons n'est pas créé à partir de rien, mais s'inspire de l'apprentissage multimodal avec les machines profondes de Boltzmann (Deep Boltzmann Machines) (Srivastava et Salakhutdinov, 2012).

2.2.3 Phase 3 : Conception du modèle d'adaptation dynamique

L'objectif principal du modèle de l'usager est de soutenir une adaptation de haute qualité tenant compte de la particularité de chaque individu. À cet effet, le modèle de l'usager permet la détection et la classification exacte de son comportement et de ses réactions. Le modèle d'adaptation doit être capable d'exploiter ce modèle de l'usager pour proposer des solutions à implémenter dans le système pour répondre convenablement aux besoins des usagers. Il doit pouvoir répondre adéquatement au profil détecté en appliquant des mesures d'adaptation, y compris l'ajustement dynamique des composants du jeu ainsi que des règles pédagogiques. Par conséquent, notre modèle d'adaptation est composé d'un moteur d'adaptation. Nous appelons notre modèle d'adaptation un modèle dynamique car il permet au système hôte (dans notre cas le jeu sérieux) de se modifier en temps réel en fonction de l'état actuel du joueur et de son comportement dans le jeu. Également, ce modèle est hybride puisqu'une partie des règles d'adaptation est apprise automatiquement à partir des données et l'autre partie est fournie manuellement par les experts du domaine.

2.2.4 Phase 4 : Intégration des modèles, test et validations

Cette phase est axée sur le test des modèles de façon abstraite individuels (ex : est ce qu'un modèle développé permet de prédire l'état des connaissances hors contexte d'application) et de façon concrète dans le jeu sérieux. Cette dernière validation ayant pour but de tester si l'adaptation atteint l'objectif escompté qui est l'apprentissage et la satisfaction de l'usager.

Ainsi, pour l'étape de validation dans le jeu, il s'agit d'abord d'intégrer dans le jeu sérieux, les modèles conçus. Le paramétrage du jeu dans cette phase consiste en :

- L'intégration d'un modèle permettant l'évaluation automatique du niveau de raisonnement socio-moral du joueur en fonction de ses justifications verbales.
- L'intégration d'un moteur d'adaptation éditable.
- La mise en place des éléments d'adaptation (musique, mouvements non verbaux des PNJs, etc.)

Ensuite nous avons conduit des expérimentations avec des participants réels afin de valider et d'ajuster les solutions proposées. La validation s'est faite en 2 étapes. Nous avons valider dans la première étape les moteurs prédictifs (le modèle usager) dans leur capacité à prédire avec précision le comportement des joueurs-apprenants dans le jeu. En second lieu, nous avons validé le moteur d'adaptation dynamique et donc sa capacité à améliorer la motivation, le gain d'apprentissage et le plaisir de jouer (*gameplay experience*). Ces validations ont donné lieu à des ajustements à faire et qui sont pris en compte dans la prochaine boucle de la méthodologie.

Quelle que soit la tâche considérée, l'évaluation suit toujours le même paradigme : celui de la comparaison des réponses du système à une référence prédéfinie (en général des données validées par les experts ou les évaluations subjectives faites par les participants). Les différentes évaluations menées dans la littérature se différencient essentiellement par le choix de la métrique de comparaison. Par exemple si l'on considère C comme étant le nombre de classifications correctes effectuées par le système et E le nombre de ses erreurs, alors parmi les métriques les plus utilisées on a :

- Le rappel qui est égal à $R = C/CR$ où CR est le nombre de classifications correctes totales (de la référence) ;
- La précision qui est égal à $P = C/(C + E)$;
- La F-mesure qui est égal à $2 \cdot P \cdot R/(P + R)$;

- Le coefficient de Kappa11 qui estime «l'accord inter-annotateur observé entre les réponses du système et la référence».

La validation de la solution d'optimisation (développée en phase 2) est faite sur des bases de données publiques. La validation des modèles prédictifs est faite sur des données recueillies dans une première expérience avec le jeu sérieux et avec Muse-logique, ainsi que sur les données à notre disposition provenant du laboratoire ABCs. La validation du modèle usager suit le protocole présenté ci-dessus. La prise en compte de différentes métriques permet d'évaluer différents angles du modèle. Les données de référence sont celles issues des formulaires d'évaluations que nous avons fait remplir aux participants après chaque partie de jeu.

En ce qui concerne l'analyse et la validation du modèle d'adaptation et donc l'efficacité du jeu, une approche commune est la comparaison des résultats avant (pré) et après (post) le jeu afin de mettre en évidence l'effet du jeu sur l'apprentissage (Eagle, 2009). De plus, des questionnaires sont utilisés pour collecter les informations démographiques et détecter les forces et faiblesses de l'utilisabilité du jeu (Froschauer *et al.*, 2010). Une autre approche plus approfondie consiste à diviser les participants en un groupe de test et en un groupe expérimental qui joue le jeu, et en un groupe témoin qui est enseigné en utilisant des méthodes traditionnelles telles que l'enseignement en classe. Dans notre cas, nous avons choisi la première approche puisque nos participants sont peu nombreux (30 participants avec la version non adaptative et 40 participants pour la version adaptative). Nous avons utilisé l'échelle de *Likert* pour la conception du questionnaire sur l'efficacité du jeu et la satisfaction du joueur. C'est une échelle psychométrique très utilisée dans les recherches utilisant des questionnaires. Les questionnaires de pré-test ont pour objectif d'évaluer les connaissances préalables du participant sur l'item de connaissance enseigné (raisonnement socio-moral pour le jeu So-Moral). Les questionnaires de post-test quant à eux permettent de déterminer la satisfaction

globale, le taux d'apprentissage et le taux d'acceptation du jeu.

2.3 Solutions attendues

Nous récapitulons ici la liste de nos contributions :

- Un nouvel algorithme (adapté à la modélisation de l'usager) pour l'optimisation de l'apprentissage dans les architectures d'apprentissage profond ;
- Un ensemble de modèles de prédictions du comportement de l'usager (cognitif, affectif, social) ;
- Un moteur d'adaptation dynamique
- Un jeu sérieux adaptatif pour l'apprentissage du raisonnement socio-moral

2.4 Conclusion

Ce chapitre avait pour objectif de présenter la méthodologie adoptée pour le développement de ce projet de recherche. Nous présentons nos solutions détaillées dans les prochains chapitres.

CHAPITRE III

UNE NOUVELLE TECHNIQUE D'ACCÉLÉRATION DES ALGORITHMES D'OPTIMISATION DE PREMIER ORDRE

L'adaptation dans les systèmes intelligents interactifs nécessite une modélisation des usagers en temps réel qui permette au système de comprendre les besoins actuels de l'usager le plus rapidement possible et d'y répondre par des actions appropriées. Ainsi, les architectures d'apprentissage profond choisies devront répondre aux exigences d'efficacité pour une mise en oeuvre aisée et adéquate de la modélisation de l'usager, tout en garantissant une certaine précision du modèle qui en découlera. Nous proposons dans ce chapitre une nouvelle technique permettant d'accélérer les algorithmes d'optimisation, utilisés dans les architectures profondes. Nous montrons que l'accélération de ces techniques permet d'améliorer les performances des modèles c'est à dire. trouver une modélisation la plus précise possible et ce, dans un temps plus restreint que les techniques existantes. Le temps étant un facteur déterminant dans un contexte d'adaptation en temps réel, produire un modèle de prédiction fiable dans un temps limité est donc un enjeu essentiel. Nous montrons que les améliorations proposées sont génériques et facilement transposables dans d'autres cadres applicatifs de l'apprentissage profond.

Nous présentons tout d'abord les techniques d'optimisation existantes pour l'apprentissage dans les architectures profondes, et mettrons en évidence leurs limites

respectives. Cet état de l'art critique sera suivi par la présentation de notre solution ainsi que la preuve de sa convergence. Nous terminerons ce chapitre par la présentation des expérimentations menées pour tester notre solution sur différentes bases de données publiques, et une étude comparative des résultats avec ceux de l'état de l'art.

3.1 Introduction

3.1.1 Motivation de la solution dans le contexte actuel : Modélisation des usagers

L'une de nos hypothèses formulées stipule que, les solutions d'apprentissage profond sont appropriées dans le domaine de la modélisation de l'usager puisqu'ils ont prouvé leur efficacité dans l'extraction des attributs cachés permettant la discrimination des données. De même, on observe une convergence des solutions de modélisation des usagers vers l'utilisation des architectures profondes, en particulier les *Deep Neural Networks* (DNN) (Tang *et al.*, 2015; Elkahky *et al.*, 2015) en français réseaux de neurones profonds, le réseau *Long short-term memory* (LSTM) en français réseau récurrent à mémoire court et long terme et le réseau *Recurrent Neural Network* (RNN) en français réseau récurrent (Wöllmer *et al.*, 2013; Piech *et al.*, 2015; Wang *et al.*, 2017; Zhu *et al.*, 2017). Cependant, une étape importante est de concrètement s'assurer ou alors améliorer leur efficacité en terme de recherche de la meilleure représentation et dans un temps restreint. Ainsi, les modèles développés doivent être capables de s'améliorer (apprentissage continu) de façon incrémentale, c'est-à-dire au fur et à mesure que de nouvelles données entrent et ce le plus rapidement possible de sorte que le nouvel état de l'usager prédit soit utilisé le plus tôt possible et de manière transparente à l'usager. Les techniques existantes peuvent être améliorées pour la modélisation de l'usager.

Il existe actuellement plusieurs approches pour améliorer l'apprentissage dans les réseaux de neurones. Nous pouvons citer entre autres l'amélioration de l'architecture dont trouver une architecture plus sophistiquée, la recherche des paramètres et hyper-paramètres optimaux, la recherche de la meilleure représentation des données, le choix du meilleur algorithme d'optimisation, etc. Ici, nous nous intéressons aux algorithmes d'optimisation du processus d'apprentissage dans les architectures de réseaux neuronaux. Les algorithmes d'optimisation dans l'apprentissage automatique en particulier dans les réseaux de neurones, visent à minimiser une fonction objective généralement appelée fonction de perte ou de coût. Cette fonction représente la différence entre les données prédites par le modèle et les valeurs attendues. La minimisation consiste donc à définir l'ensemble des paramètres communément appelés poids de l'architecture, qui donnent les meilleurs résultats dans des tâches ciblées telles que la classification, la prédiction ou le *clustering*. La recherche d'un ensemble de paramètres permettant de minimiser la fonction d'erreur dans un réseau de neurones à propagation en avant n'est pas une tâche triviale. Le processus est non seulement non linéaire mais aussi dynamique dans ce sens que tout changement d'un poids nécessite l'ajustement de nombreux autres (Eberhart et Kennedy, 1995).

Ainsi, pour que l'apprentissage de nos modèles puisse se faire rapidement, nous avons jugé nécessaire de développer notre propre technique d'accélération de l'apprentissage dans les réseaux de neurones. De plus, la solution que nous visons doit être capable de trouver une solution équivalente ou meilleure que celle trouvée par les solutions de l'état de l'art. L'avantage d'une telle solution va bien au-delà du domaine étudié dans cette thèse, le domaine étant la modélisation des usagers, puisqu'elle peut être utilisée dans n'importe quel problème d'optimisation de premier ordre utilisant la dérivée première de la fonction à minimiser.

3.1.2 Motivation de la solution dans un contexte plus général

La majorité des algorithmes d'optimisation dans les réseaux de neurones, proposé dans la littérature sont des méthodes de premier ordre (Ruder, 2016). Cependant il existe également des techniques de second ordre. Ces dernières utilisent l'estimation de la matrice hessienne (*Hessian matrix*) (LeCun *et al.*, 1993; Schaul *et al.*, 2013) qui estime les dérivées secondes de la fonction à minimiser. Ces techniques déterminent le taux d'apprentissage (*learning rate*) optimal à prendre dans des problèmes quadratiques. Elles fournissent des informations additionnelles utiles pour l'optimisation mais le calcul des dérivées secondes est très demandant en ressources machine surtout pour des modèles avec beaucoup de paramètres. De plus, les valeurs estimées sont souvent une approximation médiocre de la matrice hessienne. Ainsi, nous nous concentrerons uniquement sur les techniques de premier ordre.

Les méthodes d'optimisation de premier ordre utilisent la dérivée première de la fonction à minimiser et sont basées sur la technique de la descente du gradient (Bottou, 2010). Ces algorithmes — par exemple la propagation en arrière de l'erreur (*back propagation*) (LeCun *et al.*, 1989) — permettent de trouver une matrice de poids répondant au critère d'erreur. Adam pour *Adaptive Moment estimation* (Kingma et Ba, 2014) est probablement la solution la plus utilisée (Xu *et al.*, 2015; Goodfellow *et al.*, 2016; Isola *et al.*, 2017) dans la littérature actuellement. Cependant, il a été prouvé que Adam est incapable de converger vers la solution optimale pour un problème d'optimisation convexe simple (Reddi *et al.*, 2018). Un algorithme plus récent (AMSGrad) résout ce problème en dotant Adam et d'autres méthodes stochastiques adaptatives d'une mémoire à long terme. Malheureusement, Adam donne de meilleurs résultats que AMSGrad dans certains

cas [1]. Cela suggère que l'efficacité des algorithmes d'optimisation dépend de plusieurs paramètres entre autres de la nature des données. Par conséquent, au lieu de toujours chercher une solution absolue, qui marcherait dans tous les cas, il serait plus utile de trouver un moyen d'accélérer et donc d'améliorer tous ceux existants, puisque chacun de ces algorithmes possède ses forces et ses faiblesses selon le domaine d'application concerné.

Dans cette perspective, nous proposons une nouvelle technique visant à améliorer les performances empiriques de tout algorithme d'optimisation de premier ordre, tout en préservant leurs propriétés. Nous nous référerons à la solution appliquée à un algorithme A existant, par AA pour (*Accelarated A*). Par exemple, pour AMSGrad et Adam, les versions modifiées seront identifiées par AAMSGrad et AAdam. La solution proposée améliore la convergence de l'algorithme d'origine en trouvant un minimum d'une valeur plus basse et plus rapidement. Notre solution est principalement basée sur la variation de la direction du gradient et les mises à jour précédentes des paramètres à apprendre. Nous avons effectué plusieurs tests sur des problèmes où la forme de la fonction de perte est simple donc de forme convexe, comme la régression logistique (Rennie, 2005), mais également sur des architectures de réseaux de neurones non triviales, telles que le perceptron multicouche, les réseaux de neurones à convolution profonds (CNN) et les LSTMs. Nous avons utilisé les bases de données sur les critiques de films IMDB [2](Maas *et al.*, 2011), sur les images MNIST [3], et sur les images CIFAR-10 [4] pour valider notre solution. Il convient de mentionner que, bien que les expérimentations

1. https ://fdlm.github.io/post/amsgrad/
2. http ://www.cs.cornell.edu/people/pabo/movie-review-data/
3. http ://yann.lecun.com/exdb/mnist/)
4. https ://www.cs.toronto.edu/ kriz/cifar.html

effectuées se limitent au SGD (*Stochastic Gradient Descent*), AdaGrad, Adam et AMSGrad, la solution proposée pourrait être appliquée à d'autres algorithmes. Les résultats suggèrent que l'ajout de la solution proposée à la règle de mise à jour d'un algorithme d'optimisation de premier ordre donné améliore ses performances.

3.2 Les algorithmes d'optimisation de 1er ordre existants

3.2.1 Descente du gradient

La descente du gradient qui est une technique d'optimisation de premier ordre et celle la plus utilisée, a pour objectif de trouver les valeurs des poids qui minimisent une fonction d'erreur J(x). Cette fonction mesure la distance qui sépare les résultats prédits des résultats attendus. Une valeur proche de 0 implique que le modèle entraîné prédit correctement les sorties. Cette fonction d'erreur doit satisfaire à certaines contraintes afin de pouvoir être utilisée par l'algorithme de la descente du gradient : elle doit être écrite sous forme de moyenne, elle ne doit être dépendante que des sorties du réseau et elle doit être dérivable. À cet effet il existe plusieurs fonctions d'erreur dont l'erreur quadratique moyenne et l'entropie croisée.

3.2.1.1 Comment fonctionne l'algorithme de la descente du gradient ?

Soit une fonction $J(x)$ à variables réelles ($x \in \mathbb{R}^n$), suffisamment différentiable et dont on cherche un minimum. La descente du gradient construit de façon itérative une séquence qui en principe devrait se rapprocher d'un minimum au fur et à mesure des itérations. Pour ce faire, il commence par un point initial x_0 (une valeur aléatoire par exemple) et construit une suite récurrente comme suit :

$$x_{n+1} = x_n - \eta \cdot \nabla_{x_n} J(x) \tag{3.1}$$

Où, η est le taux d'apprentissage (*learning rate*) qui détermine la vitesse de mise à jour du pas à prendre pour atteindre le minimum et ∇ représente la gradient. Le choix de η est empirique : s'il est trop petit, le nombre d'itérations pourrait être très grand, par contre s'il est trop grand les valeurs de la séquence peuvent osciller autour d'un minimum, ayant ainsi des difficultés à converger rapidement. Néanmoins, cette méthode est assurée de converger vers un minimum avec un taux d'apprentissage bien choisi, même si les données en entrée ne sont pas linéairement séparables.

3.2.2 Algorithmes d'optimisation basés sur la descente du gradient

Comme mentionné à la section précédente, il existe plusieurs algorithmes d'optimisation se basant sur la descente du gradient qui, elle, se base sur la direction donnée par la valeur négative du gradient. Dans cette section, nous allons introduire quelques variantes populaires de l'algorithme de descente du gradient et que nous avons jugé nécessaire pour la compréhension de la solution que nous proposons.

3.2.2.1 NAG : Nesterov Accelerated Gradient

NAG (Nesterov, 1983) est une variante des solutions basées sur l'utilisation du momentum (Sutskever *et al.*, 2013). Le momentum est une méthode permettant d'accélérer le SGD en ajoutant une fraction du vecteur de mise à jour passée, au vecteur de mise à jour actuel. Ainsi, il ajoute au standard SGD, une dynamique newtonienne (Polyak, 1964). Le momentum utilise le gradient calculé au point courant et fait un grand pas dans la direction appropriée comme suit :

$$\begin{aligned} x_{n+1} &= x_n - v_n \\ v_n &= m \cdot v_{n-1} + \eta \cdot \nabla_{x_n} J(x). \end{aligned}$$

Où m est le momentum (souvent fixé à 0.9), v_{n-1} est le poids précédemment calculé et $\nabla_x J(x)$ est le gradient courant par rapport aux paramètres x, η est le taux d'apprentissage et v est appelé vélocité (vitesse dans une direction spécifique). Toutes les opérations s'appliquent élément par élément (*element-wise*). Cependant, la méthode du momentum peut facilement osciller autour des minimums locaux et peut ainsi manquer le minimum global. Nesterov est une version améliorée de cette technique : il fait un grand pas en se basant sur l'accumulation du gradient et ensuite corrige ce pas en se basant sur le gradient actuel calculé. La différence réside principalement au niveau de la position où le gradient est évalué. En effet, au lieu de calculer le gradient à la position actuelle, le NAG calcule le gradient à une position approximative de la prochaine itération puisque nous savons que la prochaine valeur des paramètres peut être estimée via $x_n - v_{n-1}$. NAG peut donc se résumer comme suit :

$$\begin{aligned} x_{n+1} &= x_n - v_n \\ v_n &= m \cdot v_{n-1} + \eta \cdot \nabla_{x_n} J(x - m \cdot v_{n-1}). \end{aligned}$$

Nesterov accélère donc le SGD classique et les autres techniques se basant sur le momentum en adaptant les mises à jour des paramètres à la dérivée de la fonction d'erreur. Toujours est-il que, le taux d'apprentissage qui est utilisé est constant pour tous les paramètres même si chacun d'eux ne se trouve pas à la même distance de leurs valeurs optimales. Peu importe si l'on se trouve proche ou éloigné du minimum recherché, le taux de mise à jour ne change pas, ce qui peut être un problème pour la convergence de cet algorithme. À cet effet, il existe d'autres techniques capables d'adapter le taux d'apprentissage en fonction des paramètres de manière à ce qu'il soit possible d'effectuer des mises à jour plus petites ou plus grandes selon l'endroit où l'on se situe sur la courbe de perte.

3.2.2.2 AdaGrad : Adaptive Gradient

Adagrad (Duchi *et al.*, 2011) est l'un des premiers algorithmes d'optimisation qui adapte le taux d'apprentissage aux différents paramètres. Le taux d'apprentissage n'est alors plus simplement un hyper-paramètre, mais il devient également un paramètre puisque en plus d'être manuellement spécifié par l'utilisateur avant l'entraînement (hyper-paramètre) il est modifié pour en trouver la valeur optimale au fur et à mesure de l'apprentissage.

Malheureusement, cet hyper-paramètre pourrait être très difficile à régler car comme mentionné ci-dessus, s'il est choisi trop petit, alors la mise à jour des paramètres sera très lente et il faudra beaucoup de temps pour obtenir un minimum acceptable. Sinon, s'il est choisi trop grand, alors les paramètres se déplaceront sur toute la fonction et pourront ne jamais atteindre un minimum acceptable. De plus, le grand nombre de dimensions et la nature non convexe de l'optimisation des réseaux de neurones peut conduire à une sensibilité différente sur chaque dimension. Le taux d'apprentissage peut être trop faible dans une dimension (paramètre) et trop élevé dans une autre. Une façon de contourner ce problème est de choisir un taux d'apprentissage différent pour chaque dimension. Dans la pratique, l'un des premiers algorithmes a avoir proposé l'adaptation du taux d'apprentissage dans les réseaux neuronaux profonds est l'algorithme AdaGrad. Cet algorithme a mis à l'échelle de façon adaptative le taux d'apprentissage pour chaque paramètre.

Ainsi, avec AdaGrad la vitesse de convergence est adaptée selon les paramètres, effectuant ainsi des mises à jour plus importantes pour les paramètres peu fréquents et des mises à jour plus petites pour les paramètres fréquents. Il s'est avéré être une solution adaptée aux problèmes d'apprentissage à grande échelle et où les données ont certaines valeurs manquantes (*sparse data*) (Dean *et al.*, 2012). Un jeu de données est considéré comme clairsemé (*sparse*) lorsque certaines valeurs

attendues dans l'ensemble de données sont manquantes, ce qui est un phénomène courant dans la fouille de données à grande échelle. C'est un problème qui altère la performance (capacité de généralisation) des algorithmes d'apprentissage machine et leur capacité à calculer des prédictions précises. AdaGrad cependant, réussit à résoudre ce problème. Sa règle de mise à jour est la suivante :

$$x_{n+1} = x_n - \frac{\eta}{\sqrt{\sum_{i=1}^{n}(\nabla_{x_i} J(x))^2 + \epsilon}} \nabla_{x_n} J(x)$$

η est la valeur initiale du taux d'apprentissage et ϵ est une petite valeur (*smoothing term*) qui permet d'éviter la division par zéro. Elle est généralement de l'ordre de 10^{-8}. AdaGrad accumule la somme au carré de tous les gradients et l'utilise pour normaliser le taux d'apprentissage afin qu'il puisse être plus ou moins grand selon le comportement des gradients passés. Un inconvénient de l'algorithme AdaGrad est que la partie d'accumulation du gradient augmente de façon monotone, ce qui est problématique car le taux d'apprentissage diminue au point que l'apprentissage s'arrête complètement à cause du très faible taux d'apprentissage. Le taux d'apprentissage aura tendance à s'approcher de 0 après un certain nombre d'itérations.

3.2.2.3 RMSprop : *Root Mean Square*

RMSprop est un algorithme d'optimisation qui n'a pas été publié dans un article de recherche, mais qui a été reçu positivement par la communauté scientifique. Il a été proposé par Geoffrey Hinton dans un de ses cours en ligne dont le titre est «*Neural Networks for Machine Learning*» [5]. RMSprop se situe dans le domaine des méthodes avec un taux d'apprentissage adaptatif comme AdaGrad. RMSprop peut être vu comme une version améliorée de AdaGrad où l'accumulation agressive des gradients au carrés qui font que AdaGrad peine à converger dans certains

5. https ://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

problèmes a été résolue.

Ainsi, pour éviter que le taux d'apprentissage ne diminue drastiquement et ne ralentisse l'apprentissage, RMSprop décompose le gradient accumulé de telle sorte que seule une partie des gradients passés est considérée. Au lieu de considérer tous les gradients passés, RMSprop se comporte comme une moyenne mobile (plus connue sous le terme : *moving average*). La méthode utilise une moyenne mobile exponentielle γ avec un taux de décroissance qui appartient à l'intervalle $[0, 1]$ (la valeur suggérée étant 0.99). L'idée centrale est de conserver la moyenne mobile des gradients carrés pour chaque poids et ensuite, de diviser le gradient par la racine carrée du carré moyen. C'est pourquoi on l'appelle RMSprop (*Root Mean Square*). Avec les équations mathématiques, la règle de mise à jour ressemble à ceci :

$$\begin{aligned} x_{n+1} &= x_n - \frac{\eta}{\sqrt{G_n + \epsilon}} \nabla_{x_n} J(x) \\ G_n &= \gamma \cdot G_{n-1} + (1 - \gamma) \cdot x_i J(x)^2. \end{aligned}$$

La mise à jour prend la portion γ de la somme cumulée passée du gradient au carré et prend la portion $(1 - \gamma)$ du gradient au carré courant. Comme on peut le voir dans la 1ère équation ci-dessus, le taux d'apprentissage est adapté en divisant par la racine du gradient quadratique, mais comme nous n'avons que l'estimation du gradient sur le mini batch actuel, RMSprop utilise la moyenne mobile de celui-ci. L'inconvénient majeur de RMSprop est que le deuxième estimateur de moment (variance excentrique) du gradient est un biais positif important au début de la descente du gradient (Kingma et Ba, 2014).

3.2.2.4 AdaDelta

AdaDelta (Zeiler, 2012) tout comme RMSprop, est une extension d'AdaGrad, qui cherche à réduire le taux d'apprentissage monotone et agressif. Il s'agit d'une

méthode d'optimisation de premier ordre mais qui a certaines propriétés des méthodes de second ordre. En effet, il fait une certaine approximation de la méthode Hessienne, qui ne coûte qu'un seul calcul de gradient par itération, en s'appuyant sur les informations des mises à jour passées. C'est aussi une méthode dont le taux d'apprentissage s'adapte en fonction des paramètres. Il s'adapte dynamiquement dans le temps en n'utilisant que des informations de premier ordre. La règle de mise à jour est la suivante :

$$\begin{aligned} x_{n+1} &= x_n - \Delta x_n \\ \Delta x_n &= \eta \sqrt{\frac{P_{n-1} + \epsilon}{G_n + \epsilon}} \\ G_n &= \gamma \cdot G_{n-1} + (1 - \gamma) \cdot \nabla_{x_n} J(\theta)^2 \\ P_n &= \gamma \cdot P_{n-1} + (1 - \gamma) \cdot \Delta x_n^2. \end{aligned}$$

Le plus grand avantage d'AdaDelta est qu'il n'est pas nécessaire de spécifier manuellement le taux d'apprentissage initial. Cependant, l'exécution d'AdaDelta est plus lente, mais les minimas locaux trouvés sont meilleurs dans la plupart des cas. De plus, sa vitesse de convergence dépend du taux d'apprentissage initial puisqu'un mauvais choix entraîne un apprentissage plus lent car il passera plus de temps à se stabiliser. Il n'est parfois pas aussi bon que des techniques dont le choix du taux d'apprentissage initial a été fait soigneusement.

3.2.2.5 Adam : Adaptive Moment Estimation

Adam (Kingma et Ba, 2014) est un algorithme d'optimisation de premier ordre de fonctions objectives stochastiques, basé sur des estimations adaptatives de moments d'ordre inférieur. Un moment est une quantité qui mesure la forme d'une fonction. Le premier moment (moyenne), une fois normalisé par le second moment (variance non centrée), donne la direction de la mise à jour. Les mises à jour d'Adam sont directement estimées à l'aide d'une moyenne mobile du premier et

du deuxième moment du gradient. Il calcule des taux d'apprentissage adaptatifs pour chaque paramètre. En plus de stocker une moyenne exponentiellement décroissante des gradients carrés passés v_n, Adam conserve également une moyenne exponentiellement décroissante des gradients passés m_n, comme la technique du momentum. Adam peut être vu comme une combinaison de RMSprop et des techniques de momentum comme NAG. Il possède une fonction de correction de biais qui l'aide à surpasser légèrement les autres techniques avec optimisation adaptative, lorsque les gradients deviennent plus sparses (Ruder, 2016). Sa formule de mise à jour des paramètres se présente comme suit :

$$\begin{aligned} x_{n+1} &= x_n - \frac{\eta}{\sqrt{\hat{v_n} + \epsilon}} \hat{m}_n \\ m_n &= \beta_1 \cdot m_{n-1} + (1 - \beta_1) \cdot \nabla_{x_n} J(x) \\ v_n &= \beta_2 \cdot v_{n-1} + (1 - \beta_2) \cdot \nabla_{x_n} J(x)^2 \\ \hat{m}_n &= \frac{m_n}{1 - \beta_1^n} \\ \hat{v_n} &= \frac{v_n}{1 - \beta_2^n}. \end{aligned}$$

Comme on peut le voir à travers les équations, Adam a deux composantes principales : une composante momentum m_n et une composante qui corresponds au taux d'apprentissage adaptatif v_n. Il existe plusieurs techniques permettant d'améliorer substantiellement Adam telles que la technique du *weight decay* (Loshchilov et Hutter, 2017), l'utilisation du signe du gradient dans des problèmes d'apprentissage distribués (Bernstein *et al.*, 2018), le changement de Adam vers le SGD au cours de l'apprentissage (Keskar et Socher, 2017) ou encore la combinaison de Adam et Nesterov (NAdam (Dozat, 2016)). NAdam (*Nesterov Adam*) s'est avéré être une variante efficace de Adam, utilisant le même principe que NAG. Afin d'incorporer NAG à Adam, les auteurs ont modifié le terme momentum d'Adam

qui est m_n. La règle de mise à jour est la suivante :

$$x_{n+1} = x_n - \frac{\eta}{\sqrt{\hat{v_n} + \epsilon}} \left(\beta_1 \hat{m_n} + \frac{(1-\beta_1)\nabla_{x_n} J(x)}{1-\beta_1^n} \right) \tag{3.2}$$

avec v_n, m_n, β_1, β_2, les mêmes paramètres qu'Adam.

Cependant, il a été récemment prouvé qu'Adam, en plus de présenter quelques erreurs dans la preuve de sa convergence, est incapable de converger vers la solution optimale pour une optimisation convexe simple (Reddi *et al.*, 2018).

3.2.2.6 AMSGrad

Un algorithme plus récent (AMSGrad) (Reddi *et al.*, 2019) corrige ce problème en dotant les techniques adaptatives d'une mémoire à long terme. La principale différence entre AMSGrad et Adam est que AMSGrad maintient le maximum de tous les v_n qui est une moyenne exponentiellement décroissante des gradients carrés passés, jusqu'à l'instant courant et utilise cette valeur maximale pour normaliser la moyenne courante du gradient. La publication originale (Reddi *et al.*, 2019) affirme qu'AMSGrad se comporte de la même manière ou mieux qu'Adam sur certains problèmes couramment utilisés dans l'apprentissage machine. La règle de mise à jour d'AMSGrad est la suivante :

$$\begin{aligned}
x_{n+1} &= x_n - \frac{\alpha}{\sqrt{\hat{v_n} + \epsilon}} \hat{m_n} \\
m_n &= \beta_1 \cdot m_{n-1} + (1-\beta_1) \cdot \nabla_{x_n} J(x) \\
v_n &= \beta_2 \cdot v_{n-1} + (1-\beta_2) \cdot \nabla_{x_n} J(x)^2 \\
\hat{m_n} &= \frac{m_n}{1-\beta_1^n} \\
\hat{v_n} &= max(\hat{v}_{n-1}, \frac{v_n}{1-\beta_2^n}).
\end{aligned}$$

Rappelons que le but d'un algorithme d'optimisation est de chercher les para-

mètres qui minimisent une fonction, sachant qu'il n'y a aucune information sur ce à quoi ressemble cette fonction. Si nous connaissions à l'avance la forme de la fonction à minimiser, il serait alors facile de faire des mises à jour plus précises des paramètres qui mèneraient ainsi au minimum escompté. Bien que nous ne pouvons pas savoir à l'avance la forme de la fonction, chaque fois que nous faisons un pas en utilisant un algorithme d'optimisation quelconque, nous pouvons savoir si nous avons passé un minimum en calculant le produit du gradient actuel et du gradient passé, et vérifier si celui-ci est négatif. Ceci nous renseigne sur la courbure de la surface de la fonction de perte et est — à notre connaissance — la seule information précise et disponible en temps réel sur la tendance de la courbe à minimiser. La méthode proposée exploite cette information pour améliorer la convergence des algorithmes d'optimisation.

En regardant de près les algorithmes présentés ci-dessus, aucune de ces techniques existantes n'utilise la variation de la direction du gradient comme information pour calculer la prochaine mise à jour. La solution proposée n'est pas spécifique à un algorithme d'optimisation particulier. Elle peut également être appliquée sur une version accélérée de l'algorithme déjà proposée, ce qui le rend facilement adaptable à toute solution.

3.3 Une méthode pour accélérer les algorithmes d'optimisation de 1er ordre

3.3.1 Intuition et pseudocode

L'idée à garder en tête est qu'on veut trouver des valeurs qui minimisent une fonction (que l'on supposera convexe) stochastique à plusieurs variables. Une solution évidente serait de dériver la fonction (∇f) et résoudre l'équation $\nabla f = 0$ (qui permettrait de trouver le minimum exact). Cependant, cette équation est à plusieurs inconnues et la résolution n'est pas triviale. C'est pour cette raison qu'on va

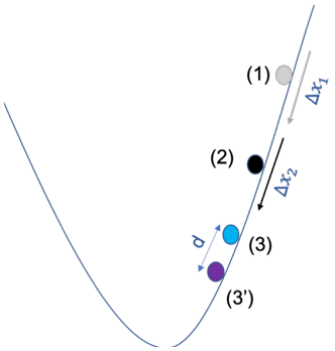


Figure 3.1 Intuition de la méthode d'accélération proposée. (1) Position initiale de la balle ; (2) Position de la balle après la première itération du GD (Gradient Descent) ; (3) Position de la balle après la 2eme itération du GD ; (3') Position de la balle après la 2eme itération du AGD (Accelerated GD). $d = -\eta \cdot \nabla f(x_1)$ est le gain de taille de pas apporté par AGD au pas pris par GD.

vers les méthodes permettant de faire une approximation du minimum, à l'instar de la descente du gradient. Si nous savions quelle était l'allure de la fonction à minimiser à l'avance il devrait être plus facile de la minimiser. Nous n'avons accès à aucune information sur l'allure générale de cette fonction. Néanmoins, nous avons accès aux directions des pentes (qui nous informent sur l'allure partielle de la courbe). L'idée de notre solution est donc d'utiliser cette information sur la variation de la direction des pentes pour accélérer le pas. Le principe étant le suivant : Si la direction de la pente ne change pas, alors on peut prendre de plus grands pas pour se diriger plus rapidement vers le minimum. Par contre, si la direction change, alors on doit continuer la descente du gradient normalement (appliquer l'algorithme de base). Nous allons expliquer notre solution de manière visuelle.

Supposons un fossé (voir figure 3.1) dont notre objectif est de diriger une balle se trouvant à une certaine position vers le point le plus bas de ce fossé. Ceci doit se faire sachant qu'il fait noir, donc on ne voit pas le point le plus bas et on ne connaît d'ailleurs pas notre position dans le fossé.

Objectif : Accélérer la descente de la balle vers le point le plus bas, sachant une visibilité limitée.

- Position initiale de la balle : x_1 ;
- Position à la prochaine itération : $x_2 = x_1 - \eta \cdot \nabla f(x_1)$ en utilisant la version la plus basique de l'algorithme de la descente du gradient ;

— Position à la 2eme itération : $x_3 = x_2 - \eta \cdot \nabla f(x_2)$;

- (3') Position à la 2eme itération avec accélération vu que on a $\nabla f(x_2) \cdot \nabla f(x_1) > 0$: $x_3 = x_2 - \eta \cdot (\nabla f(x_2) + \nabla f(x_1))$;
- gain de $d = -\eta \cdot \nabla f(x_1)$;
- Dans le cas de Adam et AMSGrad, la direction du pas passée est conservée dans le moment d'ordre 1 (m) ;
- Le test de la variation de la direction sera donc fait entre m et le gradient actuel.
- Il faut fixer un seuil $S <\mid \nabla f(x_2) + \nabla(x_1) \mid$ au-delà duquel on arrête l'accélération pour éviter d'osciller autour du minimum.

Plus le seuil S est élevé, plus nous nous rapprocherons de l'algorithme original. Toutefois, S ne doit pas être trop petit (voir section 3.5.3 de ce chapitre). Une autre solution pourrait être d'arrêter d'accélérer la balle et de laisser l'algorithme d'optimisation original prendre le contrôle total de l'apprentissage après le premier changement de direction. Cette solution fonctionne, mais converge plus lentement. Ici, nous nous concentrerons uniquement sur le cas où le gradient g_t est remplacé par $g_t + m_{t-1}$ ou $g_t + g_{t-1}$ lors du calcul du nouveau m_t dans Adam et AAMSGrad (voir algorithme 3.1 pour détails) et le calcul du nouveau g_t dans les autres algorithmes d'optimisation (SGD et AdaGrad par exemple) si la direction ne change pas. Noter que g_t peut aussi être remplacé par $g_t + v_{t-1}$ lors du calcul de v_t. Ce changement que nous avons apporté à l'algorithme d'optimisation ne change pas sa convergence puisque la quantité Γ_t (Reddi *et al.*, 2018) — qui mesure essentiellement le changement de l'inverse du taux d'apprentissage de la méthode

pseudocode : 3.1 *Version accélérée - Méthode adpatative générique*

Entrées : $x_1 \in$ F, taux d'apprentissage $(\alpha_t > 0)_{t=1}^{T}$, sequence de fonctions $(\varphi_t, \psi_t)_{t=1}^{T}$
$t \leftarrow 1$
$S \leftarrow$ seuil
repeter
  $g_t \leftarrow \nabla f_t(x_t)$
  $m_t \leftarrow \varphi_t(g_1, ..., g_t)$
  $V_t \leftarrow \psi_t(g_1, ..., g_t)$
  si $g_t \cdot m_t > 0$ et $|m_t - g_t| > S$ **alors**
    $gm_t = g_t + m_t$
    $m_t = \varphi_t(gm_1, ..., gm_t)$
  $\hat{x}_{t+1} = x_t - \alpha_t m_t V_t^{-1/2}$
  $x_{t+1} = \prod_{F,\sqrt{V_t}}(\hat{x}_{t+1})$
  $t \leftarrow t + 1$
jusqu'à t > T

adaptative par rapport au temps — garde sa propriété. Nous avons fourni en annexe une preuve théorique que la méthode que nous proposons ne change pas la limite du regret pour AAMSGrad. Notez que cette technique peut être appliquée à n'importe quel algorithme d'optimisation de premier ordre, comme on peut le voir dans l'algorithme 3.2.

pseudocode : 3.2 *ASGD*

$t \leftarrow 1$
$S \leftarrow$ seuil
$\alpha \leftarrow$ taux d'apprentissage
tant que $t < T$ **faire**
  $g_t \leftarrow \nabla_{x_t} J(x)$
  si $g_t \cdot g_{t-1} > 0$ et $\mid g_{t-1} - g_t \mid > S$ **alors**
    $x_{t+1} \leftarrow x_t - \alpha \cdot (g_t + g_{t-1})$
  sinon
    $x_{t+1} \leftarrow x_t - \alpha \cdot g_t$
  $t \leftarrow t + 1$

3.3.2 Analyse de la convergence

Nous supposons que si nous sommes en mesure de prouver que la modification d'un algorithme d'optimisation avec la méthode proposée ne modifie pas sa convergence, alors il en va de même pour les autres algorithmes d'optimisation. Ainsi, nous n'analyserons que la convergence d'un algorithme d'optimisation déterministe dont le AGD (*Accelerated Gradient Descent*) et d'un algorithme non déterministe dont le AAMSGrad.

Pour les méthodes déterministes telles que le GD, l'analyse de la convergence consiste simplement à trouver une limite supérieure de la différence entre la valeur de la fonction f à la t-ème itération et la valeur minimale $(f(x_t) - f(x^*))$, où f est la fonction à minimiser, x^* est la solution optimale et x_t est la solution trouvée par l'algorithme au temps t.

Théorème 3.3.1. *Soit $f : \mathbb{R}^d \to \mathbb{R}$ une fonction convexe de L-Lipschitz et $x^* = \text{argmin}_x f(x)$. Ainsi, le GD avec un pas (taux d'apprentissage) de taille $\alpha \leq 1/L$ satisfait à l'inéquation suivante :*

$$\begin{aligned} \sum_{t=0}^{T-1} (f(x_{t+1}) - f(x^*)) &\leq \frac{L}{2} [\|x_0 - x^*\|^2 - \|x_T - x^*\|^2] \\ f(x_T) - f(x^*) &\leq \frac{L}{2T} \|x_0 - x^*\|^2 \end{aligned} \tag{3.3}$$

En particulier, $\frac{L}{\epsilon} \|x_0 - x^*\|^2$ itérations suffisent pour trouver une valeur ϵ-approximative optimal x. Le GD a une convergence sous-linéaire de taux $O(1/T)$ [6].

Pour AGD, puisque $\nabla f(x_{T-1}) = k \nabla f(x_T)$ if $\nabla f(x_{T-1}) \cdot \nabla f(x_T) > 0$ avec $k > 0$ nous avons :

---

6. https ://www.cs.cmu.edu/ggordon/10725-F12/slides/05-gd-revisited.pdf

Théorème 3.3.2. *soit $f : \mathbb{R}^d \to \mathbb{R}$ une fonction convexe de L-Lipschitz et $x^* = argmin_x f(x)$. Ainsi, l'algorithme AGD avec un pas de taille $\alpha \leq 1/L$ satisfait à l'inéquation suivante :*

$$f(x_T) - f(x^*) \leq \frac{L(1+k)}{2T} \|x_0 - x^*\|^2 \tag{3.4}$$

L'algorithme AGD à une convergence sous-linéaire de taux $O(1/T)$ qui est similaire au GD. Le méthode d'accélération n'altère donc pas la convergence de l'algorithme de base.

Pour l'algorithme AAMSGrad, l'analyse diffère légèrement de celle du GD mais l'objectif sous-jacent reste le même. Nous avons utilisé le cadre d'apprentissage en ligne (*online learning framework*) (Zinkevich, 2003). Ainsi, la convergence pour AAMSGrad est calculée par rapport au *regret* puisque les valeurs attendues des gradients de l'objectif stochastique sont difficiles à calculer. Le *regret* est défini comme suit :

$$R(T) = \sum_{t=1}^{T} [f_t(x_t) - f_t(x^*)] \tag{3.5}$$

Notation : Pour tout vecteur $a, b \in \mathbb{R}^d$, les opérations $\frac{a}{b}$, $\sqrt{a}$, a^2, $max(a, b)$ sont effectuées élément par élément. Pour tout vecteur $\theta_i \in \mathbb{R}^d$, $\theta_{i,j}$ représente le $j$ème coordonnée où $j \in [d]$. Finalement, nous disons que $\mathcal{F}$ a un diamètre borné D_∞ si $\|x - y\|_\infty \leq D_\infty$ pour tout $x, y \in \mathcal{F}$.

Théorème 3.3.3. *Supposons que $\mathcal{F}$ ait un diamètre borné D_∞ et $\|\bigtriangledown f_t(x)\|_\infty \leq G_\infty$ pour tout $t \in [T]$ et $x \in \mathcal{F}$. Avec $\alpha_t = \frac{\alpha}{\sqrt{T}}$, AAMSGrad a un regret borné*

comme suit, pour tout $T \geq 1$ *:*

$$R_T \leq \frac{D_\infty^2 \sqrt{T}}{2\alpha(1-\beta_1)} \sum_{i=1}^{d} (\hat{v}_{T,i}^{-1/2}) + \frac{D_\infty^2}{4(1-\beta_1)} \sum_{t=1}^{T} \sum_{i=1}^{d} \frac{\beta_{1t} \hat{v}_{t,i}^{1/2}}{\alpha_t} + \frac{2\alpha\sqrt{1+\log(T)}}{(1-\beta_1)^2(1-\gamma)\sqrt{(1-\beta_2)}} \sum_{i=1}^{d} \|g_{1:T,i}\|_2 \tag{3.6}$$

La preuve de cette borne est donnée en annexe A. Le résultat suivant est le corollaire immédiat du résultat précédent :

Corollaire 3.3.3.1. *Soit* $\beta_{1t} = \beta_1 \lambda^{t-1}, \lambda \in (0,1)$ *dans le théorème 3.3.3, ainsi nous avons :*

$$R_T \leq \frac{D_\infty^2 \sqrt{T}}{2\alpha(1-\beta_1)} \sum_{i=1}^{d} (\hat{v}_{T,i}^{-1/2}) + \frac{\beta_1 D_\infty^2 G_\infty}{4(1-\beta_1)(1-\lambda)^2} + \frac{2\alpha\sqrt{1+\log(T)}}{(1-\beta_1)^2(1-\gamma)\sqrt{(1-\beta_2)}} \sum_{i=1}^{d} \|g_{1:T,i}\|_2 \tag{3.7}$$

Le *regret* de *AAMSGRAD* peut être borné par $O(2G_\infty\sqrt{T}) \simeq O(G_\infty\sqrt{T})$. Le terme $\sum_{i=1}^{d} \|g_{1:T,i}\|_2$ peut également être borné par $O(G_\infty\sqrt{T})$ puisque $\sum_{i=1}^{d} \|g_{1:T,i}\|_2 << dG_\infty\sqrt{T}$ (Kingma et Ba, 2014).

3.4 Expériences

Afin d'évaluer empiriquement la solution proposée, nous avons comparé différents modèles/architectures : la régression logistique, le perceptron multicouche (MLP), les réseaux neuronaux convolutifs (CNN) et le LSTM. Nous avons utilisé la même initialisation des paramètres pour tous les modèles précités : β_1 a été fixé à 0.9 et β_2 a été fixé à 0.999 comme recommandé pour Adam (Kingma et Ba, 2014). Le taux d'apprentissage a été fixé à 0.001 pour Adam, AAdam, AMSGrad, AAM-

SGrad et 0.01 pour les autres algorithmes d'optimisation. Les algorithmes ont été implémentés en utilisant Keras [7] et les expériences ont été réalisées en utilisant les modèles Keras intégrés. Les valeurs initiales des paramètres w à apprendre, sont restées les mêmes pour tous les modèles. Pour les versions accélérées AAdam et AMSGrad, nous ne montrons que les variantes où la valeur de m_t a été modifiée selon l'algorithme 3.1. Cependant, les deux solutions ($m_{t-1} + g_t$ ou $g_{t-1} + g_t$) ont bien fonctionné pendant nos expériences. La valeur de v_t est resté inchangée. Les expériences ont été répétées 10 fois et nous présentons les performances moyennées. Ainsi, chaque point dans les figures suivantes représente le résultat moyen après 10 entraînements sur une *epoch* (un passage sur l'ensemble des données). Le nombre d'epochs varie de 10 à 30 pour toutes les expériences car risque de surapprentissage si le nombre d'epochs est plus élevé. Pour AMSGrad, nous n'avons considéré que la version dans laquelle v_t est calculé en utilisant le maximum, et non la version utilisant la diagonale. Le code complet de notre solution est disponible sur Github [8] en version utilisable soit sous Tensorflow soit sous Keras.

3.4.1 Problèmes basiques

Nous avons d'abord testé la méthode proposée sur deux fonctions de base : une fonction convexe x^2 dont le minimum global est égal à 0 et une fonction non convexe x^3 dont le minimum n'existe pas et tend vers $\rightarrow -\infty$. Comme nous pouvons le constater dans la figure 3.2, les versions accélérées atteignent un meilleur minimum par rapport aux algorithmes originaux dans les deux cas de figure.

---

7. https ://keras.io/

8. https ://github.com/angetato/Custom-Optimizer-on-Keras

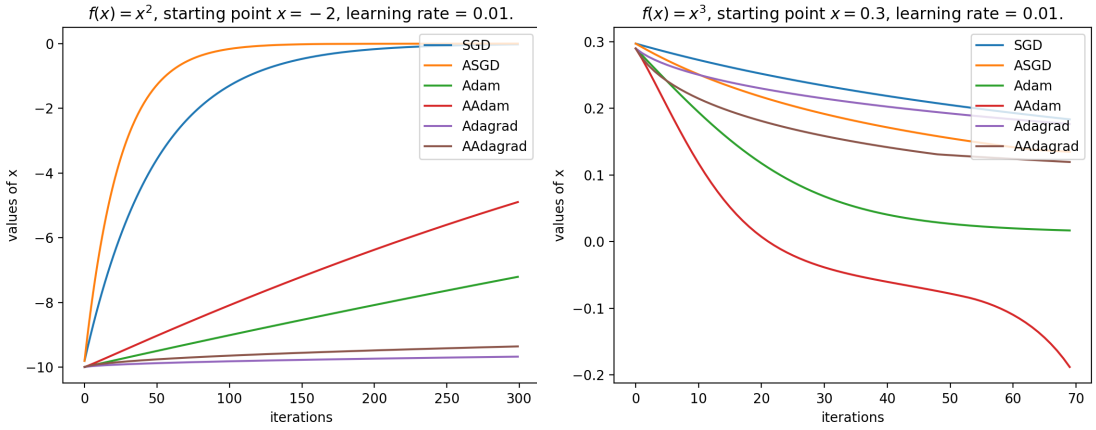


Figure 3.2 Trouver le minimum des fonctions de base avec comme points de départ $x = -10$ pour x^2 et $x = 0.3$ pour x^3. Pour AAdam, nous avons utilisé g_{t-1} au lieu de m_{t-1} dans la condition *if-else* et la mise à jour des paramètres. Aucun seuil n'a été fixé.

3.4.2 Reconnaissance d'images : MNIST

Aucun prétraitement n'a été effectué sur l'ensemble des images MNIST [9] . Tous les algorithmes d'optimisation ont été entraînés avec des mini-batchs de taille 128. Tous les poids w_i ont été initialisés à partir de valeurs tronquées en utilisant une distribution normale avec un écart-type de 0.1. Nous avons utilisé la fonction d'activation '*softmax cross entropy*'. Encore une fois, pour *AAMSGrad*, nous avons utilisé g_{t-1} au lieu de m_{t-1} dans la condition '*if-else*' et dans la formule de calcul de la mise à jour des paramètres.

La régression logistique a une fonction objectif convexe bien connue, ce qui la rend appropriée pour la comparaison de différents algorithmes d'optimisation sans se soucier des problèmes de minimas locaux (Kingma et Ba, 2014). Nous avons implémenté le même modèle présenté dans l'article original sur Adam, pour la comparaison avec d'autres algorithmes. Dans la figure 3.3 (graphe à gauche), nous

9. http ://yann.lecun.com/exdb/mnist/

pouvons voir que les versions accélérées surpassent les algorithmes originaux du début jusqu'à la fin de l'apprentissage. Nous pouvons conclure que dans les problèmes convexes simples, notre solution permet d'améliorer la convergence des algorithmes et permet ainsi de trouver un minimum plus bas. Le perceptron mul-

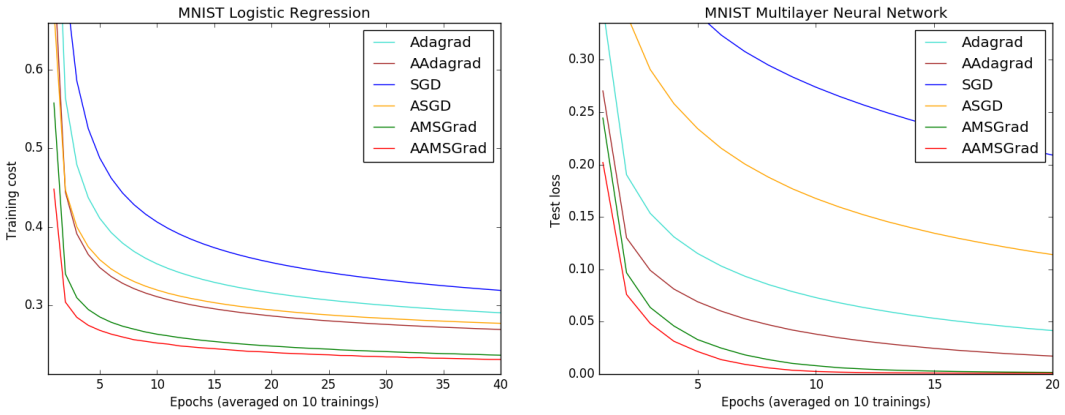


Figure 3.3 Régression logistique avec comme fonction de coût la *negative log likelihood* sur les images MNIST (à gauche). Un perceptron multicouche (MLP) sur les images MNIST (à droite). Variation dans le temps des valeurs de la fonction de coût.

ticouche (MLP) est un modèle avec des fonctions objectifs non convexes. Son architecture consiste en une couche d'entrée, une couche de sortie et n couches cachées où n >= 1. Dans nos expériences, nous avons sélectionné des architectures/modèles qui étaient conformes aux publications existantes dans ce domaine. Un réseau de neurones simple constitué d'une seule couche cachée de 512 neurones et d'une fonction d'activation (ReLU) a été utilisé pour cette expérience. La taille du *mini-batch* a été également fixée à 128. La figure 3.3 (graphe à droite) montre les résultats de l'exécution de notre modèle MLP sur les données MNIST. Encore une fois, les algorithmes accélérés surpassent (atteignent un minimum plus bas et convergent plus rapidement pendant l'apprentissage) les algorithmes originaux.

3.4.3 Reconnaissance d'images : CIFAR-10

Les réseaux de neurones convolutifs (CNNs) (LeCun *et al.*, 1995) sont des réseaux neuronaux où les couches représentent des filtres convolutifs (Krizhevsky *et al.*, 2012) appliqués aux entités locales. Les CNNs sont maintenant utilisés dans presque toutes les tâches de l'apprentissage machine (classification des textes (Tato *et al.*, 2017b), analyse de la parole (Abdel-Hamid *et al.*, 2014), etc.). Nous avons considéré le problème de classification multi-classes de l'ensemble de données standard CIFAR-10, qui se compose de 60 000 images étiquetées et de taille 32 × 32 chacun. Nous avons utilisé un réseau neuronal convolutif avec plusieurs couches de convolution et de *pooling* (en français «mise en commun», permet la réduction de la taille des images, tout en préservant leurs caractéristiques importantes). En particulier, la première architecture que nous avons testée et dont les résultats sont présentés dans la figure 3.4 graphe à gauche, contient 4 couches de convolution avec 32 canaux pour les deux premières couches et 64 canaux pour les deux autres. Le noyau de chacune des 4 couches de convolution a une taille de 3 × 3 (taille des filtres) suivie d'une couche entièrement connectée (*fully connected layer*) de taille 212. Les couches de pooling sont toutes de taille 2 × 2 et nous avons utilisé le max pooling (on ne garde que la valeur maximale des features se trouvant dans les matrices 2 × 2). Les couches de *dropout* ont été ajoutées aux 2 architectures avec des probabilités de conservation de 0.25, 0.25 et 0.5, respectivement appliquées 1) entre les deux premières couches de convolution et entre les deux couches de convolution suivantes ; 2) entre les deux dernières couches de convolution et entre la dernière couche de convolution et la couche aplatie (*flatten layer*) ; et 3) entre les couches complètement connectées (*fully connected layers*). La taille du mini-batch a été fixée à 128 comme pour des expériences précédentes. Le code de cette architecture [10] provient de la bibliothèque Keras. Les résultats

10. https ://github.com/keras-team/keras/blob/master/examples/cifar10_cnn.py

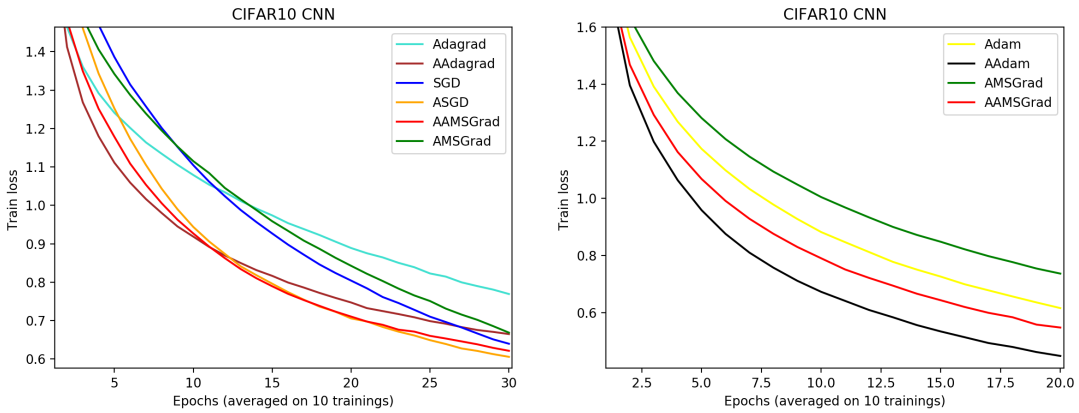


Figure 3.4 Réseaux de neurones convolutifs sur la base de données *cifar10*. Évolution de la valeur de la fonction de perte après 30 (gauche) and 20 epochs (droite). Les architectures sont c32-c32-c64-c64 (gauche) et c32-c64-c128-c128 (droite).

de cette expérience sont présentés à la figure 3.4. Comme nous pouvons le voir, les algorithmes accélérés ont donné de bien meilleurs résultats que les algorithmes originaux avec les deux architectures différentes.

3.4.4 Classification de reviews de films : IMDB

Enfin, nous avons évalué notre méthode sur les données des critiques (sous format textuel) de films d'*imdb*[11]. Cet ensemble de données est composé de 25 000 critiques de films, étiquetées par sentiment (positif/négatif). Les critiques ont été prétraitées et codées sous forme de séquence d'index de mots. Nous avons ensuite entraîné un modèle de régression logistique et un modèle LSTM (*Long Short Term Memory*). Le LSTM (que nous allons détailler dans le chapitre suivant) est devenu une composante centrale du traitement automatique du langage naturel (NLP — *Natural Language Processing*) (Tai *et al.*, 2015). L'architecture du LSTM (Hochreiter et Schmidhuber, 1997) lui permet d'apprendre les dépendances

11. https ://keras.io/datasets/#imdb-movie-reviews-sentiment-classification

à long terme en utilisant une cellule mémoire qui est capable de préserver l'information (états) sur de longues périodes. Nous avons utilisé une architecture LSTM déjà implémentée dans *Keras*. Cette architecture comprends 32 couches cachées (units), une couche de dropout récurrent (*recurrent dropout*) avec probabilité de 0.2 et une couche *dropout* simple avec probabilité de 0.2. Une couche d'*embedding* a précédé le LSTM.

La taille des vecteurs d'embedding des caractères a été fixée à 32 et la taille du batch a été fixée à 32. Nous avons considéré les 5 000 premiers mots les plus fréquents dans le vocabulaire. Les résultats sont présentés dans la figure 3.5. Comme attendu, les versions accélérées ont constamment surpassé les algorithmes originaux. Cependant, pour Adam et AMSGrad, les résultats n'étaient pas très clairs dans le cas de la régression logistique (graphe à gauche), ce qui peut être dû à la nature creuse de la matrice de données. Ce résultat implique de faire d'autres analyses sur des problèmes avec matrice creuse. Dans la modélisation de l'usager, les données creuses sont assez rares puisque cela ne se produit généralement que s'il y a très très peu d'usagers qui ont effectué certaines tâches du système.

3.5 Discussions

3.5.1 Analyse du temps d'exécution

La technique que nous proposons ajoute aux algorithmes d'optimisation d'origine, une condition qui peut augmenter le temps d'entraînement du modèle. Afin de vérifier que le temps d'exécution ainsi augmenté n'est pas un enjeu, nous avons conduit une petite expérimentation dont le but est de vérifier si ce gain de temps est compensé par la rapidité de la convergence. Que se passe-t-il si nous donnons donc un même temps d'exécution à tous les algorithmes ? Dans la figure 3.6, nous avons réalisé deux expériences où nous avons évalué la valeur minimale qu'a

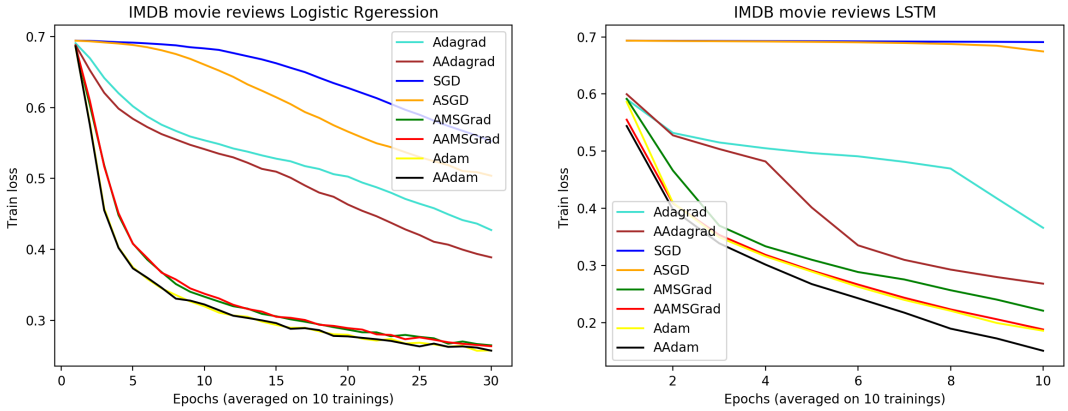


Figure 3.5 Évolution des valeurs de la fonction de perte pendant l'entraînement des modèles. Régression logistique sur les critiques de films IMDB avec comme fonction de perte la *negative log likelihood* (à gauche). LSTM sur les même données avec comme fonction de perte la *negative log likelihood* également (à droite). Aucun seuil S n'a été fixé lors de l'entraînement du modèle LSTM.

pu atteindre chaque algorithme d'optimisation sachant un temps fixe. Pour la fonction x^2, nous avons fixé le point de départ à $x = 2$ et le temps d'exécution à $t = 0.0005s$ (puisque la convergence est assez rapide pour ce problème). Comme on peut le voir dans la figure 3.6 (graphe à gauche), les versions accélérées ont fait moins d'itérations que les originaux mais ont atteint un minimum plus bas avec le même temps d'exécution. Nous avons également effectué un test avec la régression logistique sur les données MINST avec un temps fixé de $t = 10s$. Encore une fois, comme nous pouvons le voir à la figure 3.6 (graphe à droite), les versions accélérées ont largement surpassé les algorithmes d'origine. Ainsi, même si la méthode prend plus de temps par rapport aux algorithmes originaux, elle est capable d'atteindre un minimum plus bas quand on lui donne le même temps d'exécution que les autres.

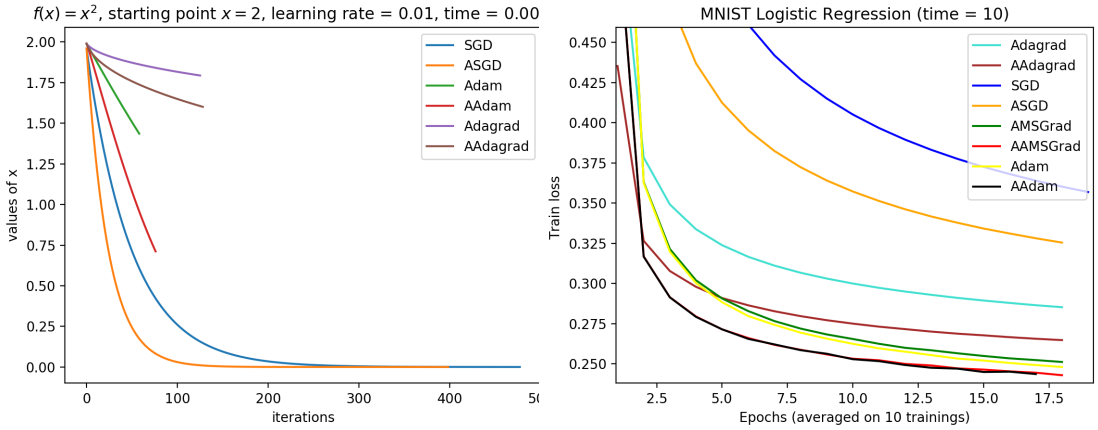


Figure 3.6 Analyse de la convergence sachant le temps d'apprentissage fixée manuellement à l'avance.

3.5.2 Adam vs AAdam

Selon le point de départ de l'algorithme (valeurs initiales des paramètres avant entraînement), Adam peut atteindre un minimum qui n'est pas optimal ou osciller autour du minimum (comme on peut le voir dans la figure 3.7 — graphe à gauche). Ceci est principalement dû au fait que Γ_{t+1} peut potentiellement être indéfini pour t $\in$[T] (Reddi *et al.*, 2018) sachant Γ définit comme suit :

$$\Gamma_{t+1} = \left(\frac{\sqrt{V_{t+1}}}{\alpha_{t+1}} - \frac{\sqrt{V_t}}{\alpha_t} \right) \tag{3.8}$$

Cependant, cette quantité reste négative pour les algorithmes n'utilisant pas les moyennes mobiles exponentielles tels que le SGD ou AdaGrad. Il est à noter que AMSGrad résout ce problème spécifique. Puisque la technique que nous proposons ne fait qu'amplifier le pas que prendrait un algorithme original, AAdam ne résout donc pas le problème d'oscillation et se comporte comme Adam mais de façon amplifiée (comme le montre la figure 3.7). Ainsi, selon le point de départ, AAdam peut atteindre un minimum d'une valeur plus basse et plus rapidement, ou seulement se comporter comme Adam (c'est-à-dire osciller autour d'un minimum).

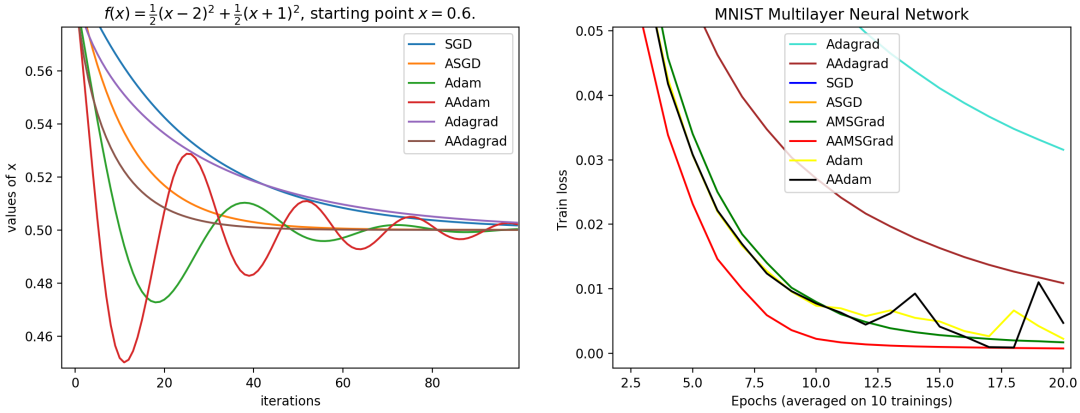


Figure 3.7 Comparaison entre Adam et AAdam.

3.5.3 Comment choisir la valeur du seuil S ?

Le choix du paramètre S est très important pour assurer l'efficacité de l'approche proposée. Dans la figure 3.8, nous avons testé différentes valeurs de S sur la régression logistique en utilisant les données MNIST. Comme on peut le voir, lorsque la valeur du paramètre est élevée, les versions accélérées se comportent pratiquement comme les algorithmes d'optimisation d'origine. Lorsque la valeur est trop petite, les versions accélérées convergent plus rapidement au début de l'entraînement mais n'atteignent pas un minimum plus bas. Il est à noter que, pour AAMSGrad, S devrait être plus petit que celui défini pour AAdam puisque le v_t utilisé pour calculer la taille du pas de mise à jour des paramètres dans AMSGrad est supérieur (valeur maximale) que dans Adam. Nous en profitons pour mentionner que durant toutes nos expériences avec les algorithmes SGD et AdaGrad, nous n'avons pas fixé de valeur pour S. La valeur de S n'a été spécifiée que pour AAdam et AAMSGrad. Les travaux futurs seront axés sur une analyse plus détaillée de l'incidence du paramètre S afin d'être à mesure de faire des recommandations sur les valeurs

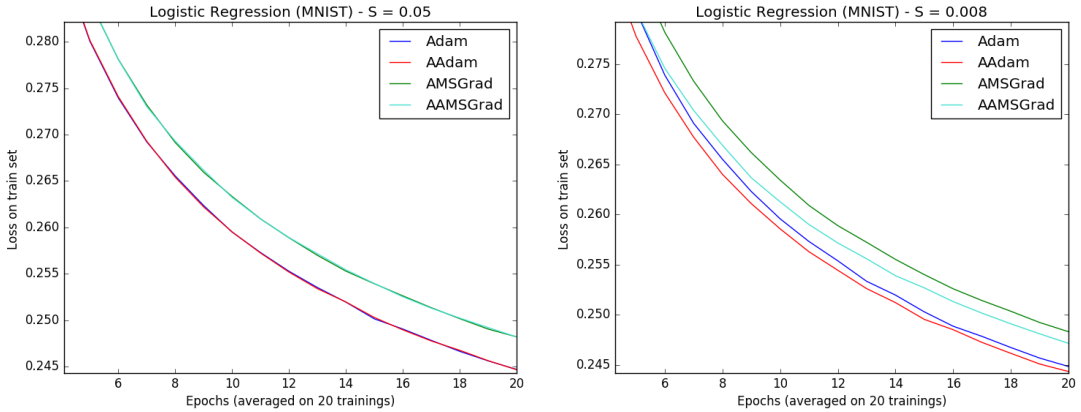


Figure 3.8 Expériences sur la variation de la valeur du seuil S.

empiriques idéales (de référence) pour ce nouveau paramètre sachant un modèle et un algorithme précis.

Dans l'ensemble, comme nous l'avons déjà indiqué, la technique proposée peut être facilement appliquée à d'autres types d'algorithmes d'optimisation de premier ordre. Si nous prenons l'exemple du NAG nous aurons ceci :

$$\begin{aligned} x_{n+1} &= y_n - \eta \cdot (\nabla J(y_n) + \nabla J(y_{n-1})) \\ &\text{si} \qquad \nabla J(y_n) * \nabla J(y_{n-1}) > 0 \\ y_{n+1} &= x_{n+1} + \beta(x_{n+1} - x_n) \end{aligned} \tag{3.9}$$

Au cours de toutes les expériences que nous avons conduites, nous avons remarqué que Adam a donné de meilleurs résultats que AMSGrad. Ainsi, la performance d'un algorithme d'optimisation dépend des données (forme de la fonction de coût), des valeurs initiales des paramètres, etc. Cette conclusion est intéressante dans le sens où la technique que nous proposons pourrait être appliquée pour améliorer n'importe quel algorithme d'optimisation, car le «meilleur» algorithme dépendra toujours de la tâche à effectuer et de la nature des données disponibles.

3.6 Conclusion

Dans ce chapitre, nous avons présenté une méthode simple et intuitive qui modifie la règle de mise à jour de tout algorithme d'optimisation de premier ordre de façon à améliorer sa convergence. Cette nouvelle solution d'optimisation est plus adaptée au problème de modélisation de l'usager dont les solutions utilisent les modèles de l'apprentissage profond puisqu'elle permet d'accélérer l'apprentissage et trouver une meilleure solution. Le temps et la précision étant deux métriques importantes dans la modélisation des usagers en temps réel.

Chaque algorithme d'optimisation existant a ses avantages et ses inconvénients. La solution que nous avons présentée a le potentiel d'accélérer la convergence de tout algorithme d'optimisation de premier ordre en se basant sur la variation de la direction du gradient ou du moment d'ordre 1 (m_t pour les méthodes à moyenne mobile). Au lieu d'utiliser uniquement le gradient pour le calcul de la mise à jour des paramètres, nous utilisons la somme du gradient courant et du gradient passé lorsque ces deux valeurs ont le même signe (même direction), sinon on garde le gradient courant. Nous avons effectué plusieurs expériences avec quatre algorithmes d'optimisation bien connus (SGD, AdaGrad, Adam et AMSGrad) sur différentes architectures profondes et en utilisant trois ensembles de données publiques (MNIST, CIFRA-10 et IMDB). Dans toutes nos expériences, les versions accélérées ont mieux performé que les versions originales, ceci tant en matière de précision de l'apprentissage que du temps de convergence. Dans le pire des cas, les versions améliorées avaient la même convergence que les versions originales, ce qui suggère que les versions accélérées étaient au moins aussi bonnes que les originaux. La technique proposée permet d'améliorer la convergence des algorithmes d'optimisation sans toutefois augmenter la complexité. Le seul inconvénient de la solution proposée est qu'elle nécessite un peu plus de calculs que l'approche stan-

dard (car elle doit calculer l'instruction *if-else* en plus), et implique de spécifier un nouveau paramètre S. Cependant, nous avons constaté que le temps supplémentaire pris par la technique était compensé par le minimum d'une valeur plus basse atteint avec un même temps d'entraînement. La nouvelle règle de mise à jour des paramètres dépend uniquement de la variation de la direction du gradient, ce qui signifie qu'elle peut être utilisée dans tout autre algorithme d'optimisation de premier ordre pour le même but. Pour terminer, la technique est intuitive, simple à mettre en œuvre et se prête bien au domaine de la modélisation de l'usager.

Le chapitre suivant sera dédié à la présentation des solutions d'architectures d'apprentissage profond proposées pour la modélisation de l'usager, qui utilisent cette technique d'accélération dans le processus d'apprentissage.

CHAPITRE IV

NOUVELLES TECHNIQUES ET ARCHITECTURES POUR LA MODÉLISATION DU JOUEUR-APPRENANT

La modélisation de l'usager peut être basée sur les attributs définis par les experts mais également par des attributs latents extraits à partir des données, c'est à dire en utilisant les techniques d'apprentissage machine. Peu de travaux se sont concentrés sur cette dernière solution car dans un premier temps, les algorithmes d'apprentissage machine existants n'étaient pas encore performants comme aujourd'hui et de plus, les données d'interactions n'étaient pas disponibles en quantité suffisante pour permettre la construction de modèles fiables. L'idée ici est donc d'utiliser ces nouvelles solutions, en particulier l'apprentissage profond, pour extraire automatiquement une représentation latente des usagers à partir des données. Dans le chapitre précédent, nous avons présenté une solution pour l'accélération de l'apprentissage dans les architectures d'apprentissage profond dans le but d'adapter ces solutions au domaine de la modélisation de l'usager mais également permettre l'obtention de meilleurs résultats. Dans ce chapitre, nous présentons nos solutions de modélisation de l'usager qui utilisent cette solution, ainsi que nos solutions pour une adaptation efficace. Nos solutions de modélisation de l'usager se basent principalement sur les architectures d'apprentissage profond. Ces modélisations visent à construire automatiquement une représentation latente des usagers afin de pouvoir prédire leurs états de connaissance, leurs émotions, etc.

Nous tenons à mentionner que les tests et validations des solutions présentées dans ce chapitre sont décrits au chapitre 5.

4.1 Préambule

Étant donné que ce chapitre porte sur plusieurs problématiques et solutions complexes en lien avec la modélisation de l'usager et l'adaptation, nous proposons de l'introduire par ce petit préambule qui résume les contributions décrites dans les prochaines sections. Les solutions proposées dans ce chapitre, visent à résoudre les problématiques suivantes :

- Comment tirer profit des connaissances a priori et a posteriori (ex : connaissances provenant des experts) dans une architecture d'apprentissage profond pour l'amélioration de la modélisation de l'usager ?
- Dans le domaine de l'éducation, il existe des connaissances rares dont les connaissances plus difficiles à maîtriser que d'autres et d'autres connaissances plus faciles à maîtriser que d'autres. Dans cette situation, les techniques d'apprentissage machine pour la prédiction des performances sont moins performants. Ceci étant dû au fait qu'il y aura peu de données sur ces connaissances. Ex : Si une connaissance est difficile à maîtriser, alors peu d'apprenants réussiront les exercices portant sur cette compétence, de même que pour les connaissances faciles à maîtriser où peu d'étudiants échoueront. Ainsi, les modèles réussiront difficilement à prédire une réussite pour une compétence difficile et un échec pour une compétence facile. Comment rendre ces modèles performants dans ces cas de figure ?
- Comment concevoir un modèle d'apprentissage profond capable de fusionner intelligemment des données d'utilisateurs provenant de plusieurs modalités ? Sachant que le comportement humain est multimodal.
- Comment extraire automatiquement des règles d'adaptation qui tiennent

compte du modèle de l'usager, pour la conception de systèmes adaptatifs ?. Nous proposons les solutions suivantes (respectivement) pour les problèmes énoncés ci-dessus :

- La solution proposée pour l'intégration des connaissances a priori et a posteriori dans les architectures d'apprentissage machine, en particulier les réseaux de neurones, tire profit du mécanisme d'attention dans les réseaux de neurones. L'idée est que durant l'apprentissage, le modèle prend les décisions (prédictions finales) en tenant compte de ce qu'il aura jugé intéressant dans les connaissances à priori/a postériori qui lui seront présentées durant l'entraînement.
- La solution proposée pour pallier le problème de connaissances rares réside à fois sur la solution précédente et sur la modification de la fonction de perte, fonction qui guide l'apprentissage. L'idée étant de forcer le modèle à être plus attentif lors de la prédiction de ces connaissances par rapport aux autres. Ceci est fait en attribuant un coût plus grand au modèle lorsque ce dernier fait une mauvaise prédiction sur les connaissances rares.
- La solution proposée pour la conception d'un modèle capable de tenir compte de la multimodalité du comportement humain, tient compte du fait que les données proviennent de plusieurs canaux qui ont une structure interne qui rend difficile d'ignorer qu'elles proviennent de canaux différents. Ainsi, nous proposons une fusion dite «*decision level*». Cette fusion est effectuée une fois que les informations latentes importantes ont été extraites de chacune des modalités. De plus, à cette fusion sont ajoutées des informations latentes extraites de la fusion «*caractéristiques level*» (fusion faite à partir des données brutes).
- La solution proposée pour l'extraction semi-automatique des règles consiste à utiliser un arbre de décision et un réseau de neurones pour extraire automatiquement ces règles. L'idée est que, une fois ces modèles entraînés, nous

analysons les paramètres appris pour en extraire des règles de la forme SI x ALORS y.

Nous détaillons dans la suite de ce chapitre chacune de ces propositions.

4.2 Architectures d'apprentissage profond

Nous rappelons que l'apprentissage profond est une approche d'apprentissage machine utilisant les réseaux de neurones à plusieurs couches et dont l'architecture peut varier selon le problème. Selon LeCun et al. (LeCun *et al.*, 2015), l'apprentissage profond permet à des modèles informatiques composés de multiples couches de traitement, d'apprendre des représentations de données à plusieurs niveaux d'abstraction. Nous présentons dans cette section quelques architectures neuronales qui nous serviront de base (*building-blocks*) pour la construction de nos solutions.

4.2.1 Les réseaux de neurones récurrents (RNNs)

La réflexion humaine ne se fait pas à partir de zéro. La pensée humaine a une persistance. En effet, les théories classiques de la cognition humaine sont fondées sur ce postulat et intègrent la mémoire (perceptuelle, attentionnelle, à court terme et à long terme) dans leur cycle cognitif (Anderson, 1996). Nous essayons de comprendre notre environnement en fonction de ce que l'on perçoit et ce que l'on sait, de ce que l'on a déjà vu. Les réseaux de neurones récurrents implémentent donc ce processus temporel, ce qui les diffère des autres architectures de réseaux de neurones. Ce sont des réseaux avec des boucles, permettant aux informations de persister dans le temps. Ces boucles permettent de transmettre l'information d'une séquence à une autre. Cette architecture en chaîne montre que les RNNs sont intimement liés aux séquences et aux listes. C'est l'architecture privilégiée dans le

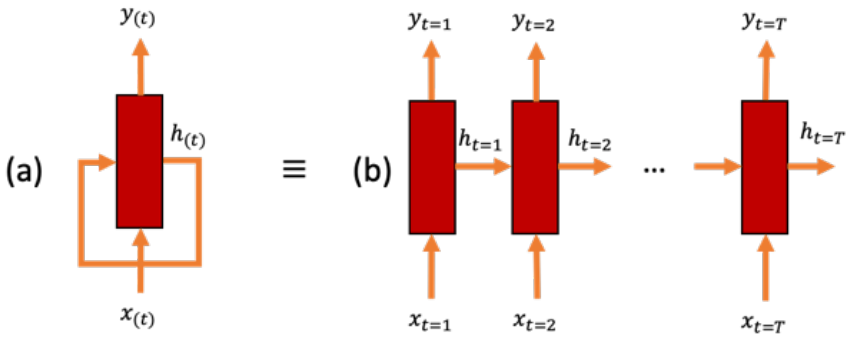


Figure 4.1 Architecture interne d'un réseau de neurones récurrents.

cas où les données sont de nature séquentielles (ex : texte), car elles permettent de prendre en compte les dépendances temporelles qui peuvent se manifester dans les données en entrée. C'est une solution qui a été utilisée avec succès dans les domaines tels que la modélisation du langage (Mikolov *et al.*, 2010), la traduction automatique (Bahdanau *et al.*, 2014), la reconnaissance de la voix (Graves *et al.*, 2013) et l'éducation (Piech *et al.*, 2015).

Ainsi, un RNN permet de transformer une suite de vecteurs de longueur T en une autre suite de vecteurs de même longueur. À chaque instant t, un même module est réutilisé. Le vecteur en sortie $y(t)$ et le vecteur caché $h(t)$ sont des fonctions de l'entrée $x(t)$ et du vecteur caché antérieur $h(t-1)$ à l'instant précédent. Un RNN est donc une structure récurrente où la sortie $y(t)$ générée à l'instant t dépend non seulement de l'entrée $x(t)$ à cet instant précis mais également de toutes les données antérieures $(x(1)...x(T))$. Cela permet à l'architecture de prendre en compte les contextes précédents. Dans la figure 4.1, la sous-figure (a) représente le RNN dont la sortie est $y(t)$ qui dépend à la fois de l'entrée $x(t)$ et des données cachées $h(t-1)$. La sous-figure (b) représente le même réseau en vue développée dans le temps, chaque bloc de RNN est en fait le même bloc avec les mêmes paramètres qui agissent à des instants successifs. Le calcul de la sortie pour un RNN simple (*vanilla RNN*) s'effectue de la manière suivante :

$$
\begin{aligned}
h_t &= \sigma(W_x \cdot x_t + W_h \cdot h_{t-1}) \\
y_t &= \theta(W_y \cdot h_t)
\end{aligned}
\tag{4.1}
$$

Avec $h_t \in \mathbb{R}^k$ représentant l'état de la couche cachée et $y_t \in \mathbb{R}^p$ l'état de la couche de sortie. En pratique on prend souvent $h_{-1} = 0$. Les paramètres $n, k, p \in \mathbb{N}$, l'ensemble $x_0 ... x_m$ avec $x_t \in \mathbb{R}^n$ est une suite d'entrées, σ et θ sont des fonctions d'activation, W_x, W_h et W_y sont les poids. Les poids W_x, W_h sont conservés dans le temps.

Bien que les RNNs soient efficaces, ils sont moins utilisés en pratique car ils ne sont pas capables de faire persister l'information après un certain temps, on parle alors de problème de *dépendance à long terme*. Ce problème est principalement dû à la rétropropagation dans le temps (BPTT — *Backpropagation Through Time* —) (Werbos *et al.*, 1990) où les signaux d'erreur qui remontent dans le temps (*backward*) ont tendance à (1) exploser ou (2) disparaître numériquement (*vanishing*) (Hochreiter et Schmidhuber, 1997). Le BPTT rencontre des difficultés de calcul numérique, puisque le gradient tend pratiquement vers 0 dans les 1ère couches cachées ce qui entraîne une mise à jour presque nulle pour les poids se trouvant à ces couches. Parfois, c'est uniquement l'information la plus récente qui a de l'intérêt pour la tâche actuelle. Les RNNs peuvent être efficaces dans ces cas (persistance à court terme). Exemple : Supposons un modèle essayant de prédire la prochaine réponse d'un étudiant sur un problème p_t portant sur une connaissance X, sachant que les 2 problèmes (p_{t-2}, p_{t-1}) qui ont précédé portaient sur cette connaissance X et que l'étudiant a répondu correctement. Le modèle n'a pratiquement pas besoin d'aller plus loin dans la séquence des problèmes répondus pour prédire une forte probabilité que l'apprenant réussisse le problème p_t.

Cependant, dans la plupart des cas réels, l'information la plus récente n'est pas suffisante pour faire la prédiction, dans ces cas le RNN est incapable d'aller chercher de l'information. Ainsi, les RNNs ne prennent en compte que des dépendances à très court terme. Le LSTM (Hochreiter et Schmidhuber, 1997) est une architecture inspirée du RNN mais augmentée d'une mémoire à long terme qui permet de gérer ce problème de persistance.

4.2.2 Réseau récurrent à mémoire court et long terme (LSTM)

Un LSTM est un RNN spécial capable d'apprendre des dépendances à long terme. C'est une architecture capable de se «souvenir» de l'information pendant de longues périodes. Dans le RNN standard, la cellule a une structure très simple qui combine l'entrée actuelle x_t et l'information cachée (h_{t-1}) et le passe dans une fonction d'activation pour produire le résultat (comme présenté ci-dessus). Le LSTM a un mode de fonctionnement un peu plus complexe puisqu'il est capable entre autres de supprimer ou d'ajouter de l'information en fonction du besoin. Son architecture de base est présentée à la figure 4.2. Il est à noter que, les blocs du

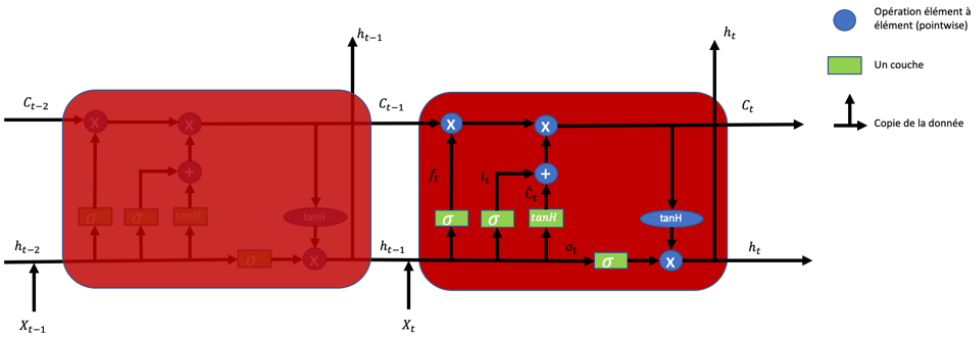


Figure 4.2 Architecture interne d'un réseau Long Short Term Memory.

LSTM (en rouge) sont semblables à ceux du RNN mais avec une architecture interne différente. Dans la figure 4.2, le C_t est appelé état cellulaire ou simplement cellule du LSTM. Elle représente la mémoire de l'architecture. Voici comment

fonctionne un LSTM :

Étape 1 : Le LSTM décide de l'information qui doit être sauvegardée ou détruite.

- Grâce au «*forget gate layer*» $f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$, il décide par rapport à l'information provenant de l'instant antérieure (h_{t-1}) et de la donnée à l'instant t (x_t) quelle information n'est pas pertinente ;
- Il regarde ensuite les deux entrées, et produit un nombre entre 0 et 1 pour chaque élément dans la mémoire C_t. La valeur «1» correspond à l'action «conserver» alors que la valeur «0» correspond à l'action «se débarrasser».

Étape 2 : Le LSTM décide de la nouvelle information à ajouter dans la mémoire C.

- Le «*input gate layer*», représenté par la deuxième sigmoïde en commençant par la gauche (figure 4.2) $i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$, décide la valeur qui sera mise à jour dans la mémoire selon les données en entrée ;
- La fonction tanh (en vert) crée le vecteur des valeurs possibles à ajouter dans la mémoire $C'_t = tanh(W_c[h_{t-1}, x_t] + b_c)$.

Étape 3 : La mise à jour de la mémoire s'effectue.

- Le LSTM supprime d'abord les valeurs inutiles dans la mémoire en utilisant le *forget gate* ;
- Ensuite, elle rajoute les nouvelles valeurs candidates qui proviennent de l'*input gate* et de la tanh : $C_t = f_t \cdot C_{t-1} + i_t \cdot C'_t$.

Étape 4 : Le LSTM décide de ce que sera la sortie.

- La sortie (h_t) est calculée en fonction de l'état actuelle de la mémoire et du vecteur de sortie (o_t). $o_t = \sigma(W_0[h_{t-1}, x_t] + b_o)$, $h_t = o_t * tanh(C_t)$.

L'idée générale est donc de contrôler la mémoire en gérant la lecture et l'écriture à l'aide des vecteurs f, i et o. Le LSTM apprend la manière dont il gère ses accès mémoires. L'un des inconvénients d'une architecture LSTM par rapport à celle d'un RNN est que le LSTM est plus lent lors de l'apprentissage puisque qu'elle

possède beaucoup plus de paramètres qui doivent être appris.

4.2.3 Les machines de Boltzmann restreintes (RBM)

Les machines de Boltzmann restreintes (RBM) (Hinton, 2002), qui ont été proposées par Geoffrey Hinton, sont des modèles particuliers de réseaux de neurones permettant de modéliser des distributions de probabilité sur des espaces discrets. Ils sont utilisés principalement dans l'apprentissage non supervisé : apprentissage avec les données non annotées. On peut les voir également comme une forme particulière du champ aléatoire de Markov pour lequel la fonction d'énergie est linéaire. Ils sont représentés par deux couches (voir figure 4.3 où chaque cercle représente un neurone) : une couche d'entrée (couche visible) et une couche cachée : il n'y a pas de couches supplémentaires d'où le terme «restreint» dans RBM. Elles sont généralement utilisées pour la réduction de dimensions (Taherkhani *et al.*, 2018), la représentation des données (Nguyen *et al.*, 2016), l'extraction de caractéristiques latents (Cai *et al.*, 2012) et la modélisation de sujets (*topic modeling*) (Srivastava *et al.*, 2013). Elles permettent d'extraire automatiquement des caractéristiques discriminantes cachées dans les données. Le RBM apprend à reconstruire les données par lui-même avec plusieurs passages de propagation en avant (*forward* – figure 4.3 (a)) et propagation en arrière (*backward* – figure 4.3 (b)). Les activations (sorties) de la couche cachée deviennent les entrées ; l'idée étant de pouvoir reconstruire la donnée qui a été passée en entrée dans la couche visible. Les poids w restent les mêmes dans les deux sens. Le résultat après les deux passages représente une reconstruction des données en entrée. L'équilibre est atteint lorsque les reconstructions sont presque semblables aux entrées initiales. Le résultat se trouve alors au niveau de la couche cachée.

Le but de l'apprentissage dans les RBMs, comme dans toutes architectures de

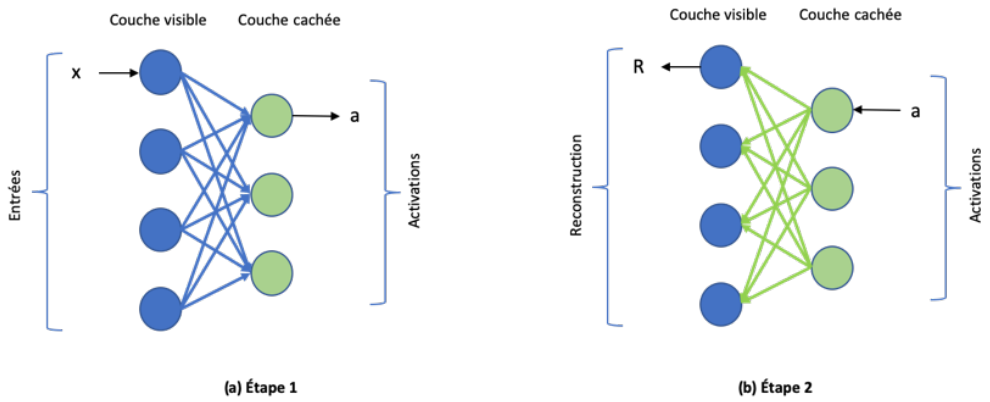


Figure 4.3 Architecture interne d'une machine restreinte de Boltzmann. Activation f ((poid w * entrée x) + biais b) = sortie a.

réseaux de neurones, est de minimiser la différence entre la reconstruction produite et l'entrée réelle en trouvant les poids par rétropropagation du gradient de l'erreur, qui minimisent cette erreur. On peut avoir plusieurs couches de RBM, à ce moment on parlera plutôt de *deep-belief networks* (DBN) (Hinton, 2009).

4.2.4 L'auto-encodeur

L'auto-encodeur (voir figure 4.4) a la même utilité qu'un RBM : trouver une représentation plus compacte de la donnée (Hinton et Salakhutdinov, 2006) ou en extraire des attributs latents. La différence entre ces deux solutions réside principalement au niveau de l'architecture et de l'algorithme d'apprentissage. Dans l'auto-encodeur, les sorties sont égales aux entrées, et le résultat qui nous intéresse se trouve dans la couche du milieu. L'idée étant d'entraîner le réseau à reproduire leur entrée tout en minimisant l'erreur de reconstruction. Ainsi, l'auto-encodeur essaie d'apprendre la fonction identité $f(x) = x$ où f est la fonction que le modèle essaie de reproduire. Cependant, il doit essayer de reproduire la fonction identité tout en ayant comme contrainte le fait que la couche cachée du réseau (couche du milieu) doit avoir un nombre restreint de neurones sinon, l'apprentissage serait

trivial. L'auto-encodeur peut aussi être vu comme un modèle de compression de la donnée. Par exemple si on prend une image de taille 10×10 alors l'entrée est de taille 100 ; si la couche cachée contient 25 neurones, alors l'image sera réduite en une image de taille 5×5. Le réseau sera ainsi forcé d'apprendre une représentation compressée de la donnée en entrée. Il est le plus souvent utilisé dans les cas réels comparativement au RBM. Ceci est probablement dû fait que les auto-encodeurs sont en quelque sorte plus faciles à entraîner et à utiliser que les RBMs mais donnent les mêmes résultats.

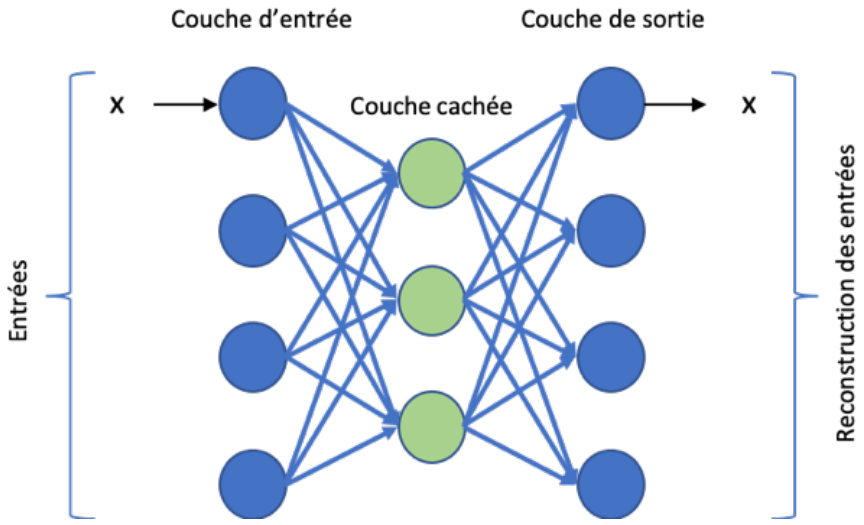


Figure 4.4 Architecture interne d'un auto-encodeur

4.3 Modélisation de l'usager à partir des techniques d'apprentissage profond

L'utilisation des solutions d'apprentissage machine vise à apprendre automatiquement de nouvelles caractéristiques qui pourraient être intéressantes dans la modélisation et pour l'adaptation. L'objectif étant de prendre en compte les aspects de l'usager et les spécificités des données du système non modélisés par les approches théoriques.

Certaines théories nous guident sur les caractéristiques à observer chez les usagers (ex : OCC (Ortony *et al.*, 1990), les styles d'apprentissage). Ces attributs qui sont déterminés *manuellement* sont importants pour la caractérisation de l'usager car ils fournissent des informations sémantiques de haut niveau qui peuvent guider la caractérisation de différents profils d'usager. Cependant, l'élicitation manuelle des attributs est subjective et les attributs potentiellement utiles et donc discriminatoires, peuvent être ignorés. En effet, les attributs ayant un sens pour les humains ne correspondent pas nécessairement à l'espace des attributs détectables et discriminants. Par ailleurs, les profils théoriques identifiés doivent être concrètement valides, c'est-à-dire observables ou tout au moins déductibles dans la réalité. D'où le besoin d'utiliser les données d'interactions pour extraire des caractéristiques objectives et pertinentes sur l'usager. Ces caractéristiques cachées dans les modèles appris, serviront à la prédiction de comportements. Dans ce dernier cas, on parlera plutôt de caractéristiques latentes ou cachées car elles ne sont pas directement observables dans les données.

Nous tenons à mentionner ici que lorsque nous parlons de modèle de l'usager, nous faisons référence aux modèles d'apprentissage profond entraînés et capable de faire des prédictions sur les usagers. Un modèle qui représente au mieux un usager est capable de prédire ses comportements. Ce modèle peut être constitué d'une seule ou de plusieurs architectures neuronales. Nous présentons dans ce qui suit les différents modèles proposés.

4.3.1 Réseaux de neurones hybrides

Un réseau de neurones artificiel est une unité de calcul élémentaire qui prend une donnée en entrée, la multiplie par un poids et l'applique à une fonction d'activation pour enfin retourner un résultat. Les approches utilisant les réseaux de neurones

sont dites guidées par les données, puisque l'apprentissage et le modèle extrait dépendent uniquement des données d'entraînement. Ainsi, leur performance dépend fortement de la quantité et de la qualité des données. En général, l'utilisation des approches entièrement axées sur les données nécessite l'acquisition d'une énorme quantité de données, ce qui n'est pas toujours pratique ou réaliste pour des raisons économiques ou en raison de la complexité du processus qu'elle implique. Il existe plusieurs domaines dans lesquels il n'y a pas suffisamment de données d'entraînement mais dont les connaissances a priori et a posteriori sont disponibles. Notamment dans les domaines de l'éducation et des sciences humaines. Comment faire profiter une architecture purement guidée par les données mais efficace, de cette connaissance souvent acquise pendant plusieurs années de recherches ? De telles solutions combinant deux approches différentes, sont souvent appelées approches hybrides.

Towell et al (Towell et Shavlik, 1994) ont défini les techniques d'apprentissage hybrides comme «des méthodes qui utilisent la connaissance théorique d'un domaine et un ensemble d'exemples pour développer une méthode de classification précise d'exemples non vus pendant l'entraînement». Ce faisant, une approche d'apprentissage hybride devrait apprendre plus efficacement comparé aux approches qui n'utilisent qu'une seule des deux sources d'information. Il existe très peu de recherches sur la combinaison des connaissances a priori /a posteriori et des architectures profondes. Parmi ces travaux on retrouve Towell et al (Towell et Shavlik, 1994) (qui à notre connaissance est l'un des premiers travaux de recherche à s'intéresser à cette question) qui proposent un système hybride appelé KBANN (*Knowledge-Based Artificial Neural Networks*). Ils mettent en correspondance les connaissances des experts, représentées dans la logique propositionnelle, dans des réseaux neuronaux et affinent ensuite ces connaissances reformulées à l'aide de la rétropropagation. Coro et al (Coro *et al.*, 2013) ont combiné les réseaux de

neurones avec des connaissances d'experts simulées. L'expert simulé a été utilisé pour générer quelques exemples, qui ont été ensuite ajoutés à l'ensemble d'entraînement de leur modèle neuronal. Zappone et al (Zappone *et al.*, 2018) ont récemment procédé de la même façon en combinant les connaissances d'experts et les réseaux neuronaux artificiels (ANN) dans le but d'optimiser les réseaux de communication sans fil. Ils ont d'abord obtenu un ensemble d'entraînement à partir des connaissances de l'expert, puis ont entraîné l'ANN sur cet ensemble d'entraînement généré. Enfin, l'architecture pré-entraînée a été affinée grâce à une nouvelle phase d'entraînement basée sur des données réelles mesurées.

Dans le domaine de l'éducation, les connaissances a priori et a posteriori sont généralement disponibles et utilisées pour construire des systèmes tutoriels intelligents. Dans d'autres domaines, les connaissances peuvent être disponibles par le biais de livres ou de maquettes déjà construites, comme les modèles à base de règles d'inférence. Nous pensons que ces connaissances, parfois acquises durant des décennies de recherche intense, ne peuvent être simplement ignorées. Ainsi, pour notre première solution de modélisation, nous proposons une approche utilisant le mécanisme d'attention (Xu *et al.*, 2015) et qui capitalise sur la disponibilité de ces données a priori et a posteriori dans le but de réduire la quantité de données empiriques à utiliser ainsi que la complexité des architectures d'apprentissage profond. À notre connaissance, aucune recherche ne propose de combiner les connaissances à priori et a posteriori avec les architectures neuronales en utilisant le mécanisme d'attention. De plus, aucune recherche dans le domaine de l'éducation, en particulier dans le domaine de la modélisation de l'usager ne s'est intéressée à cette question malgré la disponibilité des connaissances. Nous montrons que la combinaison de connaissances et de méthodes axées sur les données à l'aide du mécanisme de l'attention constitue une solution appropriée pour la conception des réseaux de neurones profonds hybrides.

4.3.1.1 Connaissances a priori et a posteriori

Parfois, les connaissances a priori et a posteriori peuvent être disponibles, mais elles ne sont pas toujours suffisamment précises ou dans une forme facile à exploiter. Néanmoins, même des modèles imprécis peuvent fournir des informations utiles qui ne doivent pas être écartées (Zappone *et al.*, 2018).

En philosophie, les connaissances a priori sont définies comme étant des connaissances non acquises par expérience contrairement aux connaissances a posteriori (Greenberg, 2010). Dans les connaissances a priori et a posteriori, nous incluons les connaissances provenant d'experts. Les connaissances d'experts représentent les compétences démontrées par les experts dans un domaine particulier. Dans le domaine de l'éducation, les connaissances a priori et a posteriori sont généralement disponibles. Par exemple, dans (Tato *et al.*, 2017a), les connaissances provenant d'experts ont été utilisées par l'intermédiaire d'un réseau bayésien, conçu entièrement par les experts du domaine pour prédire les compétences en raisonnement logique des apprenants. Ces connaissances sont disponibles dans la majorité des cas sous forme d'écrits dont les livres, les résultats d'expériences, etc. Dans de ce dernier cas, il existe une branche de la recherche qui s'intéresse aux solutions d'élicitation (formalisation) de ces connaissances. Nous pouvons citer à cet effet les travaux de (Ford et Sterman, 1998; Cooke, 1999; Martin *et al.*, 2012).

Notre objectif étant de jumeler les connaissances a priori et a posteriori avec les techniques guidées uniquement sur les données, il faudra donc rendre ces connaissances interprétables et utilisables par une machine. Une approche bien connue consiste à construire un système à base de règles (Cordón *et al.*, 2001; Fisher *et al.*, 1990). Par exemple, Van et al (Van Melle, 1978) ont utilisé les connaissances provenant d'experts pour développer le système expert MYCIN. Cependant, il peut être difficile et fastidieux d'opérationnaliser l'expertise sous forme de

règles. C'est une tâche qui fait généralement appel à l'intervention des ingénieurs de la cognition. Également, ces expertises sont pour la plupart enfouies dans une représentation écrite. Ce que nous proposons est d'utiliser des modèles préconçus (ex : systèmes à base de règles, réseaux bayésiens, etc.) tels quels lorsque disponibles, ou alors d'utiliser des techniques usuelles de traitement du langage naturel (*Natural Language Processing*) et de recherche d'informations (*information retrieval*) pour l'extraction de connaissances dans le texte.

4.3.1.2 Mécanisme d'attention

Le mécanisme d'attention permet à un réseau de neurones de se focaliser en partie sur un sous-ensemble d'informations. Les réseaux récurrents basés sur l'attention ont été appliqués avec succès à une grande variété de tâches : traduction automatique (Bahdanau *et al.*, 2014), synthèse de l'écriture manuscrite (Graves, 2013), reconnaissance de la parole (Chorowski *et al.*, 2015). L'attention en lui-même est un vecteur de poids. Il permet aux réseaux de neurones de sélectionner de façon «intelligente» l'information contenue dans l'entrée, jugée importante pour la prise de décision. Le mécanisme d'attention se présente comme un auto-encodeur basique (voir figure 4.5 [1]). Le principe derrière ce mécanisme est de connecter un vecteur de contexte entre l'encodeur (en bleu) et le décodeur (en rouge). Le vecteur de contexte prend en entrée les sorties de toutes les cellules en entrée pour calculer la distribution de probabilité des éléments de la source pour chaque sortie que le décodeur veut générer. Ce mécanisme est similaire à l'attention humaine dont le but est de se focaliser sur une partie de l'information actuelle que l'on juge importante pour une prise de décision future. Le vecteur de contexte est construit comme suit : pour un élément cible fixe (ce qui représente un mot dans la figure 4.5), tout d'abord, une première boucle est faite sur toutes les sorties des

1. shorturl.at/gpxK6

couches cachées (états) des encodeurs pour comparer les états cible et source afin de générer des scores pour chaque état dans les encodeurs. Ensuite, la fonction *softmax* est utilisée pour normaliser tous les scores, ce qui génère la distribution de probabilité conditionnée par les états cibles. Enfin, les poids sont introduits pour faciliter l'apprentissage du vecteur de contexte. Mathématiquement, voici comment est calculé le vecteur d'attention :

$$\begin{aligned} \alpha_{t,s} &= \frac{\exp(score(h_t, \overline{h}_s))}{\sum_{s'=1}^{S} exp^{score(h_t, \overline{h_{s'}})}} \\ c_t &= \sum_s \alpha_{t,s} \cdot \overline{h}_s \\ a_t &= tanh(W_c[c_t; h_t]) \end{aligned} \tag{4.2}$$

La variable h_t représente la sortie du décodeur à l'instant t, c_t est le vecteur de contexte et a_t est le vecteur d'attention. Ici, le *score* est désignée comme une fonction basée sur le contenu. Elle permet d'évaluer à quel point chaque entrée codée ($\overline{h}_s$) correspond à la sortie actuelle h_t du décodeur. Les scores sont normalisés en utilisant la fonction *softmax* ($\alpha_{t,s}$). Ce score peut être calculé de différentes façons entre autres en utilisant le produit : $score(h_t, \overline{h}_s) = h_t^{\intercal} \cdot \overline{h}_s$ (Luong *et al.*, 2015). Finalement, la variable c_t est le vecteur de contexte.

4.3.1.3 Architecture hybride proposée

L' architecture interne des réseaux de neurones rend difficile l'incorporation de la connaissance du domaine au processus d'apprentissage (Lu *et al.*, 1996). Notre solution consiste à contraindre le modèle à prêter attention à ce que les connaissances a priori et a posteriori *pensent* de l'entrée actuelle x. Elle vise donc à incorporer la prédiction finale de l'expert dans le système, plutôt que la manière dont ces connaissances sont traitées par l'expert. Puisque le mécanisme de l'attention (Xu *et al.*, 2015) est un mécanisme d'accès à la mémoire comme présenté ci-dessus, il convient parfaitement dans ce contexte où nous souhaitons que le mo-

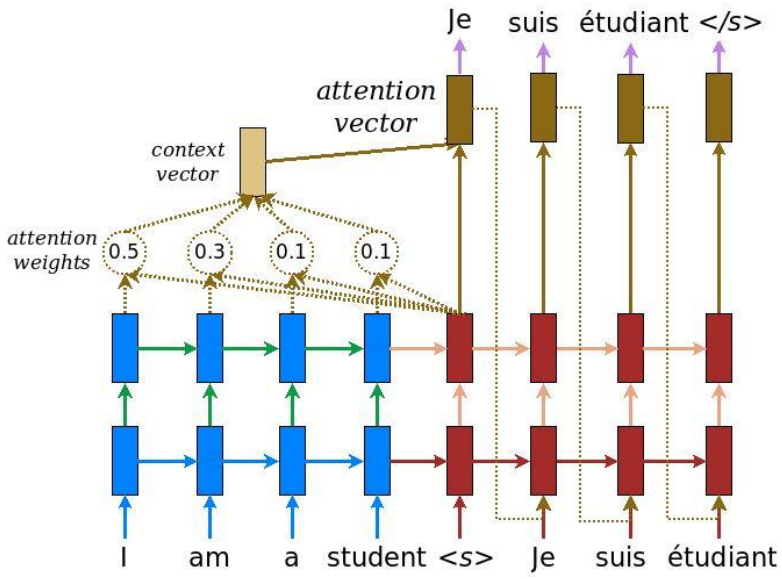


Figure 4.5 Mécanisme d'attention. Exemple de traduction de l'anglais (*I am a student*) vers le français (je suis un étudiant).

dèle ait accès aux connaissances existantes pendant l'apprentissage. En d'autres termes, le réseau de neurones «consultera» ces connaissances avant de prendre la décision finale. L'importance que le modèle neuronal accordera à ce que ces connaissances prédisent comme sortie, est calculée au moyen de pondération de l'attention (W_a et W_c, voir figure 4.6). Au fur et à mesure de l'apprentissage, le réseau saura quelle importance il accordera à chacune des prédictions provenant des connaissances en fonction des essaies/ erreurs qu'il aura commit. W_a correspond aux poids calculant l'importance de chaque caractéristique apprise par l'architecture neuronale (y_t) par rapport à chaque caractéristique extraite à partir des connaissances. W_c représente les poids mesurant l'importance des prédictions faite à partir des connaissances (via le vecteur de contexte) et des caractéristiques apprises (y_t) pour l'estimation du vecteur de prédiction finale (voir figure 4.6).

Ainsi, le modèle se concentrera sur ce que dit l'expertise dont les connaissances a priori et a posteriori, avant de prendre une décision. En fusionnant ainsi les

connaissances des experts avec l'architecture neuronale à l'aide du mécanisme d'attention, le modèle traite de façon itérative les connaissances en sélectionnant le contenu pertinent à chaque étape. Dans le mécanisme d'attention présenté par Luong et al (Luong *et al.*, 2015), spécifiquement le modèle attentionnel global, le vecteur attentionnel est calculé à partir de l'état caché cible h_t et de l'état caché en entrée. Au lieu de l'état caché d'entrée $\overline{h}_s$ comme présenté ci-dessus, nous aurons les données issues de la connaissance experte qui seront utilisées pour calculer le vecteur de contexte C_t que nous appellerons vecteur de contexte côté expert (voir Figure 2 dans (Luong *et al.*, 2015)). Ainsi, étant donné l'état caché y_t qui est la prédiction finale du modèle neuronal, et le vecteur de contexte côté expert c_t^e, nous utilisons une couche de concaténation pour combiner les informations des deux vecteurs pour produire l'état caché attentionnel a_t comme suit :

$$a_t = tanh(W_c[c_t^e; y_t]) \tag{4.3}$$

Le vecteur attentionnel a_t ainsi que la prédiction faite à partir des connaissances e et l'état caché y_t (la prédiction) du modèle neuronal sont ensuite envoyés dans une couche dense pour produire le résultat attendu y'_t. Le vecteur de contexte côté expert est alors calculé comme suit :

$$\begin{aligned} score(e_k, y_t) &= e_k \cdot y_t \cdot W_a + b \\ \alpha_{t,k} &= \frac{\exp(score(e_k, y_t))}{\sum_{j=1}^{s} \exp(score(e_j, y_t))} \\ c_t^e &= \sum_k \alpha_{t,k} \cdot e \end{aligned} \tag{4.4}$$

Où $1 \leq k \leq s$, e est la prédiction faite à partir des connaissances a priori et a posteriori, y_t est la prédiction actuelle faite par l'architecture neuronale et s est la taille du vecteur prédit (le nombre de classes à prédire). La variable e représente un vecteur de longueur égal à la taille du vecteur à prédire où chaque entrée représente la

probabilité que l'entrée appartienne à chacune des classes selon les connaissances de l'expert. La variable e_k représente de taille 1 et W_a,W_c,W_s,W_s,y_t,e, a_t sont de taille s. Le score comme mentionné ci-dessus est un vecteur basé sur le contenu qui calcule la corrélation (score d'alignement) entre les connaissances expertes et les *caractéristiques* latents appris par l'architecture neuronale. Ce paramètre définit comment les connaissances expertes et les *caractéristiques* appris à partir des données sont alignés. Le modèle attribue un score $\alpha_{t,k}$ à la paire d'entités à la position t et aux connaissances expertes (e_k, y_t), en fonction de leur correspondance. L'ensemble des $\alpha_{t,k}$ sont des pondérations définissant à quel point chaque caractéristique de la donnée provenant de l'expert doit être prise en compte pour chaque sortie (prédiction finale). La figure 4.6 montre en détail ce processus global.

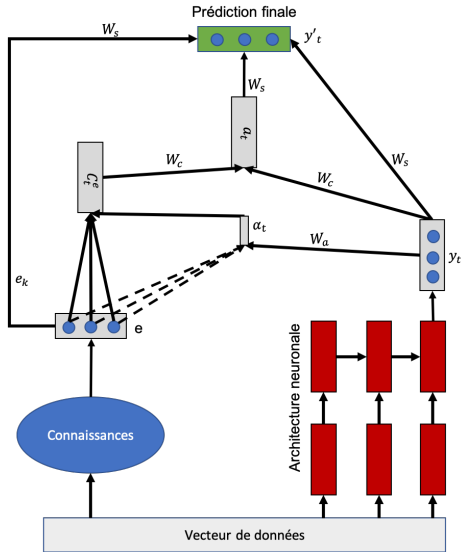


Figure 4.6 Modèle hybride attentionnel global — à chaque instant t, le modèle déduit un vecteur de poids d'alignement $\alpha_{t,k}$ basé sur les prédictions actuelles de l'architecture neuronale y_t et toutes les entrées du vecteur calculé à partir des connaissances de l'expert e. Le vecteur de contexte côté expert c_t^e est ensuite calculé comme étant la moyenne pondérée, selon $\alpha_{t,k}$, sur chaque entrée du vecteur de connaissances expert.

La figure 4.6 présente le mécanisme d'attention appliqué aux connaissances. Le

principe de fonctionnement est simple et intuitif :

- Dans un premier temps, la donnée x en entrée est traitée par l'expert c'est-à-dire, on évalue l'appartenance de x à l'une des classes à prédire à partir des connaissances a priori et a posteriori qui sont encodées de façon à être utilisée de cette manière. La même données x est également traitée par l'architecture neuronale en parallèle. Dans la figure, nous avons pris l'exemple d'un LSTM mais il peut être remplacé par un CNN ou un simple DNN.
- Ce que nous appelons «connaissances» dans la figure peut être remplacé par tout processus simulant le fonctionnement de l'expert ou tout processus simulant les connaissances a priori et a posteriori. Ça peut être un moteur à base de règles ou un réseau bayésien par exemple. La sortie des connaissances est un vecteur représentant une distribution de probabilité dont chacune des entrées est la probabilité que la donnée x appartienne à chacune des classes à prédire. Par exemple si on veut prédire qu'une image est soit un chat ou un chien, alors la sortie des connaissances est un vecteur de taille deux dont la première valeur représente la probabilité que l'image appartienne à la classe chat et la deuxième est la probabilité que l'image appartienne à la classe chien. Dans l'architecture que nous proposons, nous avons supposé que la taille du vecteur de sortie de l'expert est égale au vecteur de sortie de la prédication finale. Cependant elle peut tout aussi bien être de taille différente.
- Nous avons considéré uniquement la sortie de la dernière cellule (y_t) du LSTM, mais on pourrait aussi concaténer tous les états cachés ($h_0, ..., h_t$) pour le calcul du vecteur de contexte.
- Une fois que les vecteurs expert (e) et y_t sont estimés, l'architecture détermine le vecteur de contexte expert (appelé c^e). Ce vecteur représente l'information (les caractéristiques) pertinente extraite de la combinaison

de la connaissance experte et de l'information extraite par l'architecture neuronale. Il est calculé suivant la formule de calcul présentée ci-dessus.

- Une fois le vecteur de contexte expert estimé, le vecteur attention (a) est calculé à partir de la concaténation du vecteur de contexte expert et y_t (voir formule de calcul ci-dessus).
- Pour l'estimation du vecteur de prédiction finale y'_t, il y a plusieurs solutions possibles. La première (qui est présentée dans la figure) serait de concaténer le vecteur expert (e), la donnée apprise par l'architecture neuronale (y_t) et le vecteur d'attention a_t puis passer le vecteur ainsi obtenu dans une fonction d'activation pour calculer le vecteur final. Les autres solutions consisteraient à par exemple ne pas inclure l'un ou les deux vecteurs e et y_t à la concaténation.

Nous avons présenté notre approche en prenant pour exemple un LSTM, mais dans le cas d'un CNN, le y_t représenterait la sortie finale du réseau qui a été aplati. Le modèle hybride ainsi proposé permettra de tenir compte des connaissances a priori et a posteriori des experts pour la modélisation des usagers. L'un des usages d'un tel modèle est le traçage des connaissances ou la prédiction des compétences, que nous présenterons dans le chapitre suivant dédié à l'application de nos solutions dans des cas réels.

4.3.2 Traçage profond des connaissances rares

La modélisation de l'usager dans un système visant l'apprentissage passe avant tout par la modélisation de sa connaissance, puisque c'est cette dernière que le système vise à améliorer. L'approche la plus populaire pour modéliser la connaissance de l'usager est le *knowledge tracing* (KT) (pour traçage des connaissances). Le KT vise à modéliser la façon dont les connaissances des apprenants évoluent pendant l'apprentissage. La connaissance est modélisée sous forme d'une variable

latente et est mise à jour en fonction des performances de l'usager-apprenant au fur et à mesure qu'il effectue les tâches. Cette approche est formalisée comme suit : sachant les interactions d'un apprenant jusqu'au temps t (x_1 ... x_t) sur une tâche d'apprentissage particulière, comment va-t-il performer au temps $t+1$? L'objectif étant d'estimer la probabilité $p(x_{t+1}|x_1...x_t)$. Il existe plusieurs solutions pour estimer cette probabilité entre autres le *Bayesian Knowledge Tracing* (BKT) et le *Deep Knowledge Tracing* (DKT).

Le BKT : Il s'agit d'une approche de modélisation latente de l'apprenant par un réseau bayésien qui peut être soit entraîné à partir des données soit défini par les experts. Un réseau bayésien est un modèle probabiliste qui se présente sous forme de graphe acyclique représentant un ensemble d'éléments (nœuds) liés entre eux par des liens de causalité, dont chaque élément possède sa table de probabilité. Les réseaux bayésiens sont à la fois des modèles de représentation des connaissances et des machines à calculer les probabilités conditionnelles (Tsamardinos *et al.*, 2006). Le BKT (Corbett et Anderson, 1994) est un cas particulier des chaînes de Markov cachées où les connaissances de l'usager sont représentées par un ensemble de variables binaires : la connaissance est maîtrisée ou non. Les observations sont aussi binaires : un usager peut réussir ou échouer un problème (Yudelson *et al.*, 2013). Il ne peut pas le réussir à 30 % ou à 70% par exemple. Cependant, il existe une certaine probabilité (G, paramètre *Guess*) que l'usager donne une réponse correcte à un exercice X sachant qu'il ne maîtrise pas la connaissance liée à cet exercice. Pareillement, un usager qui maîtrise une connaissance donnera généralement une réponse correcte sur les exercices liés à cette connaissance, mais il existe une certaine probabilité (S, le paramètre *Slip*) que l'usager n'y réponde pas correctement. Le modèle standard de BKT est donc défini par quatre paramètres : l'état de connaissance initiale, la vitesse d'apprentissage (*learning parameters*), les probabilités de *Slip* et de *Guess* (*mediating parameters*). En règle générale, le BKT

utilise les connaissances à priori (L_0) sur une connaissance (T) et la probabilité d'apprendre cette connaissance ($p(T)$) pour mesurer les progrès de l'apprentissage des usagers. Il a été utilisé avec succès dans les systèmes visant l'apprentissage dans plusieurs domaine, entre autres la programmation informatique (Kasurinen et Nikula, 2009), la lecture (Beck et Chang, 2007) et le raisonnement logique (Tato *et al.*, 2016). Le BKT utilise un réseau bayésien pour capturer les connaissances des étudiants, ce qui permet de déduire la probabilité de maîtrise d'une compétence à partir d'un patron de réponses spécifiques (Conati *et al.*, 2002). La performance (échec ou réussite) est la variable observée et les connaissances sont les variables latentes. L'utilisation d'un réseau bayésien implique parfois de définir manuellement les probabilités a priori et d'étiqueter manuellement les interactions avec des concepts appropriés. De plus, les données de réponse binaires utilisées pour modéliser les connaissances, les observations et les transitions imposent une limite aux types d'exercices pouvant être modélisés. Le DKT a donc été proposé afin de résoudre les problèmes du BKT.

Le DKT : C'est une technique récente de modélisation des connaissances de l'apprenant qui utilise l'apprentissage profond. Elle s'implémente avec un RNN ou LSTM, puisque les données d'interactions sont temporelles et que cette temporalité est importante pour mieux modéliser les connaissances et être à mesure de prédire le comportement. Le principe est d'apparier une séquence d'interactions $x_1...x_T$ vers une séquence de sortie $y_1...y_T$ représentant chacune l'état de connaissances sous forme de probabilité de maîtrise, de l'apprenant après chaque interaction x. La figure 4.7 présente un RNN dont l'architecture est celle proposée à l'origine par Piech (Piech *et al.*, 2015). Ici, $h_1...h_T$ représente les encodages successifs des informations importantes passées, utiles pour les prédictions futures.

Dans le DKT, les interactions des usagers avec le système sont converties en

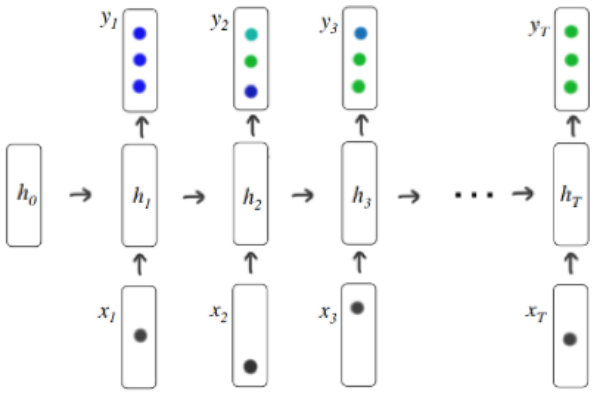


Figure 4.7 Traçage de connaissances à partir d'un réseau de neurones récurrent (Piech *et al.*, 2015). Chaque x_i représente une question portant sur une compétence particulière. Chaque y_i correspond au vecteur de probabilités où chaque valeur représente la probabilité de maîtrise d'une des connaissances en place dans le système. Au fur et à mesure que l'apprenant répond à des questions (x), le réseau met à jour l'état actuel de ses connaissances (y).

séquences de vecteurs x_t de taille fixe ; x_t étant un vecteur *one-hot* dont les valeurs sont soit 0 ou 1. Il existe à cet effet deux principales méthodes permettant de définir cette taille : quand le système contient des exercices qui portent sur un nombre limité de connaissances ou un grand nombre de connaissances. La première solution étant celle la plus utilisée. Si le système comporte des exercices ou des tâches portant sur peu de connaissances (de taille M) alors les vecteurs x_t seront de taille $2M$ chacun, où les M premières valeurs correspondent à la réponse (0 pour échec et 1 pour réussite) à un exercice portant sur l'une des M compétences. Les M dernières valeurs correspondent à l'encodage de la connaissance mis en jeu dans l'exercice courant. Par exemple, si à l'instant t c'est la connaissance $k < M$ qui est testée, alors les M premières valeurs de x_t seront toutes à 0 excepté pour la valeur se trouvant à la position k. Cette valeur sera 1 si l'exercice a été réussi et sera égale à 0 dans le cas contraire. Les M dernières valeurs seront toutes à 0 sauf la valeur à la k-ième position puisque c'est la k-ième connaissance qui est évaluée. Les vecteurs y_t sont de taille M où chaque valeur $i < M$ représente la probabilité

de réussir à un exercice portant sur la connaissance i. Ainsi le réseau se construit au fur et à mesure, l'état de connaissance latent de l'usager. Dans le cas où le nombre de connaissances (M) à apprendre est très grand, alors on utilise plutôt une distribution gaussienne pour générer des vecteurs de taille $\log M$ à partir du vecteur de taille $2M$.

L'objectif du DKT est d'entraîner un modèle RNN capable de prédire les performances futures d'un usager sachant son activité passée. Il permet intrinsèquement de définir la meilleure «séquence» d'exercices à présenter à un apprenant puisqu'il est capable de prédire l'état de connaissance escompté après une séquence d'exercices bien définie. Il est donc possible de définir cette séquence à l'avance et observer ce qui se passera et également la modifier en fonction de ce que l'on vise comme état de connaissances pour l'usager. Nous pouvons également, étant donné un état de connaissances caché estimé, interroger le DKT sur l'état de connaissances futur d'un apprenant si un exercice en particulier lui est présenté. Il permet de déterminer les corrélations pouvant exister entre les exercices (probabilité de réussir un exercice i sachant la réussite d'un exercice j) et même entre les connaissances (probabilité de réussir un exercice d'une connaissance particulière i sachant la maîtrise d'une autre connaissance j). Cette corrélation peut être donnée par :

$$J_{ij} = \frac{y(j|i))}{\sum_k y(j|k)} \tag{4.5}$$

Où $y(j|i)$ est la probabilité de correctement répondre à l'exercice j au temps $t+1$ sachant que l'apprenant a correctement répondu à l'exercice i au temps t. Cette corrélation permet également d'extraire les pré-requis associés aux exercices. L'un des avantages notables du DKT est qu'on n'a plus besoin d'experts pour définir les relations pouvant exister entre les exercices ou les connaissances, qui est une activité très coûteuse et délicate. Également, comparé au BKT dont les données d'interactions ne peuvent qu'être encodées sous format binaire, le DKT permet un

encodage qui peut varier du binaire aux valeurs réelles. Le seul inconvénient d'une telle technique, comme toutes les autres techniques d'apprentissage machine, est qu'elle a besoin d'être entraînée sur beaucoup de données représentatives pour être à mesure de bien généraliser sur les nouvelles données et de concevoir des représentations latentes assez fiables et fidèles. L'entraînement de ce modèle consiste à minimiser une fonction de coût qui est la *fonction de vraisemblance* qui s'écrit de la manière suivante :

$$L = \sum_{t} \ell(y_t^{\top} \delta(e_{t+1}), p_{t+1}) \tag{4.6}$$

Où, $\delta(e_{t+1})$ représente l'encodage *one-hot* (vecteur creux caractérisé par une composante ayant la valeur 1 et toutes les autres la valeur 0) de l'exercice répondu au temps $t+1$, ℓ est la fonction d'entropie croisée (*binary cross entropy*). Le paramètre p_{t+1} représente la réponse réelle de l'apprenant à la question $e_t + 1$, elle peut être 0 ou 1. Enfin, y représente le vecteur de connaissances prédit par le modèle. Cette fonction mesure donc la différence entre la réponse que prédit le modèle $(y_t^{\top} \cdot \delta(e_t + 1))$ pour chaque question au temps t par rapport à la réponse que fournit réellement l'apprenant (p_{t+1}) à chaque question au temps t.

4.3.2.1 Connaissances rares

Nous appelons connaissance rares, deux types de connaissances : celles difficiles à assimiler et celles faciles à assimiler. Nous appelons connaissance difficile à assimiler, une connaissance dont on observera en général peu de gens qui réussiront sur des exercices/tâches évaluant sa maîtrise (la moyenne de maîtrise de cette connaissance est faible). Nous appelons connaissance facile à assimiler, une connaissance dont presque tout le monde réussira sur des exercices/tâches évaluant sa maîtrise (la moyenne de maîtrise de cette connaissance est haute). Ce qui arrive avec ces connaissances rares est que, si on a un jeu de données sur des usagers ayant interagi avec des tâches liées à ces connaissances-là, on aura très peu de données

sur les gens qui ont bien répondu sur les exercices portant sur les connaissances difficiles et mal répondus sur les exercices portant sur les connaissances faciles. Étant donné que les techniques d'apprentissage machine (dont le DKT) se basent sur les données vues pour généraliser, nous verrons dans le chapitre 5 que ces solutions ne seront pas capables de bien généraliser sur les connaissances rares et pourront ainsi donner lieu à un modèle de traçage de connaissances inefficace dans ces cas-là.

4.3.2.2 DKT sur les connaissances rares

Compte tenu des raisons énoncées plus haut, il est important de mettre en place une solution qui améliore les techniques d'apprentissage automatique en particulier lorsqu'il s'agit du traçage de connaissances, lorsqu'il y a présence des connaissances rares. Ce problème s'apparente au problème de données déséquilibrées (*unbalanced data*) (He et Garcia, 2008) à la différence que dans ce contexte on devra faire un double balancement puisqu'on a les connaissances mais aussi les réponses associées aux exercices de chaque connaissance qu'il faut gérer. Ainsi, chaque connaissance est considérée comme une classe, mais qui à son tour est représentée par deux sous-classes (réussite d'un exercice/échec d'un exercice). Le problème de données déséquilibrées en apprentissage machine s'explique par le fait qu'il peut avoir une différence significative dans la rareté des différentes classes à prédire. Par exemple, si on veut détecter le cancer, on peut s'attendre à avoir des bases de données avec une grande quantité de données annotée comme «sans cancer» et une minorité annotée comme «avec cancer». La performance globale de n'importe quel modèle d'apprentissage machine sur ce type de données sera affectée par son habileté à prédire les données rares. Il existe dans la littérature plusieurs techniques pour pallier les données déséquilibrées entre autres le ré-échantillonnage du jeu de données (*under-sampling vs over-sampling*)(Oquab *et al.*, 2014) et le système de pondération de la fonction de coût (*cost sensitive learning*) (Shen *et al.*, 2015).

- **Undersampling (Sous-échantillonnage de la classe majoritaire)** : Le sous-échantillonnage est une stratégie simple à implémenter qui vise simplement à réduire le nombre d'exemples de la classe majoritaire afin de rééquilibrer (artificiellement) le jeu de données d'apprentissage. On choisit donc aléatoirement un certain nombre de données qui vont être extraites de la classe ayant le plus d'exemples de manière à ce que, la taille d'exemples restante soit sensiblement égale à la taille de ou des exemples dans les autres classes. L'inconvénient majeur du sous-échantillonnage est la perte d'information puisqu'on supprime volontairement des exemples qui pourraient aider le modèle à mieux généraliser.
- **Oversampling (Sur-échantillonnage de la classe minoritaire) :** C'est une stratégie similaire au sous-échantillonnage à la différence qu'elle vise plutôt la classe ayant le plus petit nombre d'exemples (classe minoritaire). L'idée est de dupliquer les exemples de la classe minoritaire afin d'obtenir une taille idéale pour le modèle. La duplication peut consister à simplement copier les exemples tels quels, mais il existe des techniques beaucoup plus sophistiquées comme la technique *SMOTE* qui au lieu de simplement dupliquer, créé de nouveaux exemples qui sont des interpolations des exemples de la classe minoritaire. Le sur-échantillonnage peut facilement entraîner un sur-apprentissage ou *overfitting* c'est-à-dire meilleur sur les exemples d'apprentissage mais moins bon sur les autres exemples (pouvoir de généralisation faible) du modèle.
- **Cost sensitive learning (Pondération de la fonction de perte) :** Ici, il s'agit principalement de pénaliser le modèle lorsqu'il fait des erreurs sur des classes spécifiques (qu'on aura identifiées manuellement). Cette stratégie vise à biaiser le modèle afin qu'il accorde plus de poids aux observations de la classe ou des classes minoritaires. En pratique il s'agit de définir une matrice de coûts de mauvaise classification. Par exemple on pénalisera le

modèle (attribuer un poids plus fort) lorsqu'il fera de mauvaises prédictions sur les exemples de la classe minoritaire. La définition de cette matrice peut être difficile dans le sens que, si les poids sont mal définis le modèle pourrait avoir tendance à privilégier la classe minoritaire et donc être moins bon dans les prédictions des exemples des classes majoritaires.

Pour résoudre le problème des connaissances rares, nous proposons une solution utilisant la pondération de la fonction de coût, sachant que les classes minoritaires sont les connaissances difficiles à maîtriser et donc très peu de bonnes réponses (majoritaires en considérant les classes de mauvaises réponses) et les classes majoritaires sont les connaissances faciles à maîtriser donc beaucoup de bonnes réponses (minoritaires en considérant les classes de mauvaises réponses). Les techniques de ré-échantillonnage peuvent tout aussi bien être utilisées mais la technique de pénalisation est plus efficace et plus simple à implémenter dans le cas du DKT.

L'amélioration du DKT que nous proposons se fait donc en 3 principales étapes :

- Identification des connaissances rares ;
- Définir les poids à attribuer pour pénaliser le modèle (pondérer la fonction de coût) ;
- Redéfinition de la fonction de perte en fonction de ces connaissances.

Nous proposons deux techniques pour identifier les connaissances rares. La première technique que nous appellerons *double boucle DKT* (*DDKT*) consiste à entraîner un premier modèle DKT sur les données, et à identifier les connaissances sur lesquelles le modèle se comporte moins bien en utilisant des métriques telle que la *F-mesure* pour chaque connaissance. La deuxième technique qui demande moins de ressources machine, consiste à conduire des tests statistiques afin de calculer pour chaque classe (connaissance) le nombre d'exercices bien répondus versus le nombre d'exercices mal répondus. Une fois ces connaissances identifiées,

on doit définir les poids de pénalisation et redéfinir la fonction de perte du DKT. L'idée principale est de traiter la fonction de perte comme une moyenne pondérée où les poids sont spécifiés par des paramètres λ_i avec $i \in [0, n]$ où n est le nombre de parties à ajouter à la fonction de perte de base. Ainsi, nous avons ajouté un terme de régularisation à la fonction de perte. Ce terme correspond à l'application d'un masque à la fonction de base dans le but d'ignorer les connaissances ayant beaucoup d'exemples (connaissances qui ne sont ni difficiles ni faciles). Le résultat du masque comporte deux parties : les connaissances très difficiles et les connaissances très faciles à maîtriser. Chaque partie du masque est multipliée par les poids λ_1 et λ_2 respectivement. En pénalisant ainsi le modèle on lui impose un coût additionnel lorsqu'il fait de mauvaises prédictions sur les connaissances rares durant l'apprentissage. Nous expliquons maintenant en détail comment obtenir la nouvelle fonction de coût.

Supposons que nous voulons évaluer le DKT sur sa capacité à prédire les réponses des exercices liés à une seule connaissance k. Alors la fonction de perte serait écrite comme ceci :

$$L_k = \sum_t \ell(y_{k,t}, p_{k,t+1}) \tag{4.7}$$

Où $y_{k,t}$ représente la probabilité qu'un apprenant réponde correctement à un exercice portant sur la connaissance k au temps $t+1$. C'est un vecteur de taille égale au nombre de connaissances, avec 0 sur toutes ses entrées sauf sur celle à la position k. Le paramètre $p_{k,t+1}$ représente la réponse (performance) réelle de l'apprenant donnée au temps $t+1$ au même problème lié à la compétence k. Le masque que nous appliquons sur la fonction de perte a pour but de diminuer le poids des connaissances avec de nombreux échantillons pour ne garder que ce sur quoi nous voulons que le modèle se concentre. Nous exécutons ensuite le modèle avec une

nouvelle fonction de perte qui s'écrit comme suit :

$$L = \sum_{t} \ell(y_t^T \delta(e_{t+1}), p_{t+1}) + \lambda_1 \sum_{t} \sum_{cD} \ell(y_{cD,t}, p_{cD,t+1,1}) + \lambda_2 \sum_{t} \sum_{cF} \ell(y_{cF,t}, p_{cF,t+1,0}) \tag{4.8}$$

Ici, λ_1 et $\lambda_2 \geq 0$ sont les poids que nous appliquons au masque et cD, cF désignent respectivement toutes les connaissances difficiles à maîtriser et toutes les connaissances faciles à maîtriser. L'indice «1» dans $p_{cD,t+1,1}$ indique les bonnes réponses et l'indice «0» dans $p_{cF,t+1,0}$ indique les mauvaises réponses . Si $\lambda = 0$, le modèle devient le DKT original. Plus la valeur de λ est élevée, plus le modèle sera biaisé vers les connaissances rares. Le choix de la valeur de λ est donc important. Les paramètres $y_{k,t}$ et $p_{c,t+1}$ sont les vecteurs y_t et p_{t+1} respectivement, où nous ne conservons que les valeurs liées à la connaissance c. Ainsi $p_{cD,t,1}$ fait référence à un vecteur de longueur égale au nombre de connaissances en jeu avec 0 sur toutes les entrées sauf pour les entrées cD et où les réponses réelles données sont correctes au temps $t+1$. Le paramètre $\delta(e_{t+1})$ est un vecteur *one-hot* qui identifie l'exercice qui est répondu au temps $t+1$. Cette nouvelle fonction de perte offre en quelque sorte un autre moyen d'équilibrer les données. Notre solution sera évaluée au chapitre suivant ou nous allons tracé les connaissances des apprenants dans le système tutoriel intelligent Muse-logique.

4.3.3 Réseaux de neurones multimodaux

L'information dans le monde réel provient de plusieurs canaux d'entrée par exemple, les images sont associées à des légendes, les vidéos contiennent des signaux visuels et audio etc. (Srivastava et Salakhutdinov, 2012). L'information étant donc de nature multimodale, il convient d'en tenir compte pour en tirer le maximum de bénéfices. De plus le comportement humain est multimodal, la perception qui mène

à l'action provient de multiple sources différentes qui sont les cinq sens. Ainsi, il est logique de tenir compte de cette multimodalité dans la conception d'une représentation de l'usager humain. Dans certains contextes, on peut n'avoir accès qu'à une seule modalité. À ce moment, les solutions présentées ci-dessus conviendraient parfaitement. Cependant, dans plusieurs autres contextes, les informations sur l'usager sont captées à partir de plusieurs canaux, entre autres ses émotions, ses mouvements oculaires, la variation de ses états cognitifs, etc. Comment mettre en commun ces données comportementales provenant de canaux différents pour en construire une représentation fidèle de l'usager ? Il est important de considérer toutes sources de données nous informant sur son état actuel.

«Chaque modalité est caractérisée par des propriétés statistiques très distinctes qui rendent difficile d'ignorer le fait qu'elles proviennent de canaux d'entrée différents» (Srivastava et Salakhutdinov, 2012). Des caractéristiques de haut niveau peuvent être apprises sur ces données en fusionnant les modalités en une représentation conjointe qui capture les spécificités auxquelles les données correspondent. Il existe dans la littérature de nombreux algorithmes d'extraction et d'apprentissage d'attributs à l'instar des réseaux mono-couches, par exemple le RBM et l'auto-encodeur. À notre connaissance, il n'existe pas d'algorithmes permettant d'extraire automatiquement des caractéristiques latentes de l'usager à partir de données multimodales provenant des activités dans les systèmes intelligents visant l'apprentissage. Il existe néanmoins quelques méthodes, techniques et outils permettant l'apprentissage d'attributs à partir de données multimodales (analyses de données affectives, classification d'images). Malheureusement, plusieurs de ces techniques essaient d'extraire les caractéristiques séparément pour chaque modalité (*decision-level* (Poria *et al.*, 2015)) et les combiner par concaténation, par moyenne ou par vote pondéré (Gunes et Piccardi, 2007; Zeng *et al.*, 2007). Le risque d'opérer de la sorte dans notre cas est que : 1) parce que les caractéristiques

d'un usager ne sont pas indépendantes, il peut y avoir des risques associés à la perte de corrélation et d'informations significatives entre les caractéristiques extraites de modalités différentes ; 2) il peut avoir un risque lié au temps d'exécution et de réponse (analyse en temps réel), car il faut considérer chaque modalité une à la fois et à tour de rôle. À cet effet, nous proposons une approche à la fois *feature-level* (dont le but est de combiner les caractéristiques extraites de chaque canal d'entrée en un «vecteur joint» (Poria *et al.*, 2015)) et *decision-level*. Le but étant de tirer le maximum d'informations possible à partir des données à disposition. De plus l'architecture multimodale proposée tient compte du fait que les données n'arrivent pas toutes en même temps et pas à la même fréquence. Il existe des architectures multimodales qui ont été proposées dans la littérature mais aucune d'entre elles n'a été conçue pour répondre à ce besoin particulier. Elles ne peuvent donc pas être utilisées pour la modélisation de l'usager. La figure 4.8 présente l'ar-

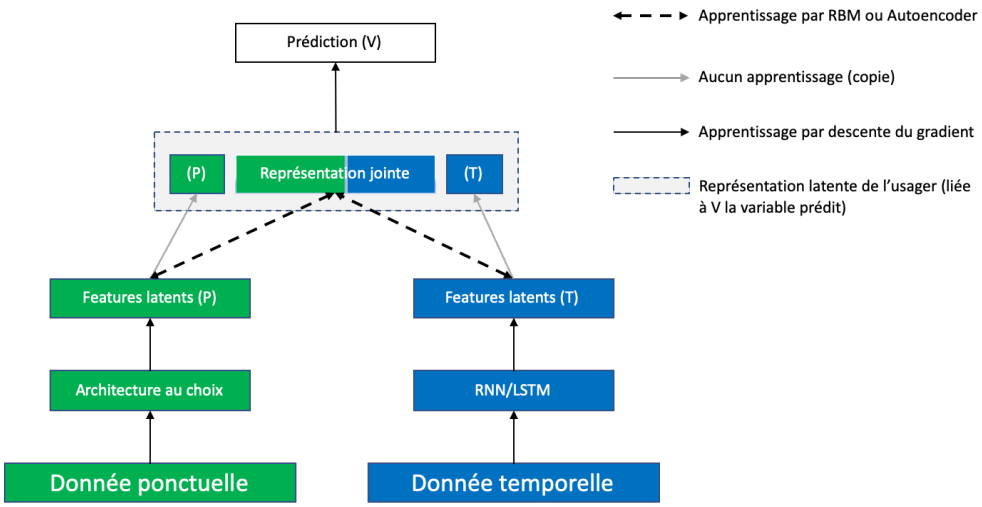


Figure 4.8 Modèle multimodal pour la modélisation de l'usager.

chitecture proposée. Ce modèle peut être utilisé pour prédire différents états de l'utilisateur selon les données en entrées. La particularité est qu'elle peut prendre un ou plusieurs types de données en entrée pour modéliser l'usager et pouvoir faire

des prédictions. Dans cette figure, les branches peuvent être rajoutées ou enlevées selon le contexte d'application et la variable v prédite. Voici comment est estimé la variable à prédire :

$$\begin{aligned} V &= f(W_f \cdot Y + b_f) \\ Y &= [P \oplus PT \oplus T] \\ PT &= RBM([P \oplus T]) \; ou \; PT = AE([P \oplus T]) \end{aligned} \tag{4.9}$$

Avec f une fonction d'activation quelconque, W_f les poids reliant les nœuds de la couche finale et l'avant-dernière couche du modèle multimodal. Y représentant la concaténation des caractéristiques extraites des données temporelles et ponctuelles séparément et des caractéristiques communs aux deux types de données (PT). Ces caractéristiques communs peuvent être apprises en utilisant un RBM ou un auto-encodeur. L'opérateur $\oplus$ signifie ici une concaténation. Les opérations se font éléments par éléments (*element-wise*). La couche précédent la couche finale est ce que nous appelons la représentation latente de l'usager. C'est en fait à partir de l'information contenue dans cette couche que la prédiction pourra se faire. Dans le chapitre 5, cette architecture est testée pour la prédiction des états émotionnels.

4.3.4 Un métamodèle pour la modélisation de l'usager

La modélisation d'un joueur-apprenant n'est pas une tâche simple surtout si on veut le représenter avec un modèle précis. Nous avons proposé différentes solutions utilisant l'apprentissage profond et permettant de développer un modèle de l'usager plus performant (comme nous allons le voir dans le chapitre 5 dédié à l'évaluation de chacune des solutions présentées ci-dessus). Ainsi, nous proposons un métamodèle qui permet aux concepteurs de créer les combinaisons de solutions qui satisfont au mieux leurs besoins. Par exemple :

- La solution d'optimisation est utilisable dans tous les modèles d'apprentissage profond, en particulier ceux visant la modélisation des usagers de systèmes intelligents ;
- On peut améliorer la prédiction de l'état de connaissances, des états émotionnels etc., en utilisant un modèle hybride qui combine les modèles d'apprentissage machine et les connaissances expertes ou théoriques ;
- On peut améliorer la prédiction de l'état de connaissance en utilisant un DKT amélioré pour les situations avec des connaissances faciles/difficiles ;
- On peut construire un modèle à partir de données multimodales (lorsque disponible) qui peut prédire par exemple l'état de connaissances, les états émotionnels etc. mais qui peut également servir à catégoriser les comportements que peuvent présenter les usagers ;
- Finalement, ces différentes solutions peuvent être combinées ou pas selon le contexte et ce que l'on veut modéliser chez les usagers.

Le métamodèle de l'usager que nous proposons est donc un ensemble de modèles d'apprentissage profond pouvant prédire différents aspects le représentant. Un modèle capable de prédire avec précision les comportements futurs d'un joueur-apprenant, est donc un modèle qui a bien appris les caractéristiques de ce joueur-apprenant au point de le comprendre et «*agir*» comme lui.

4.4 Adaptation

Les systèmes d'apprentissage adaptatifs devraient motiver et donner à l'usager l'occasion d'apprendre jusqu'à ce que les objectifs soient atteints (Kujala *et al.*, 2010). Afin de rendre un système adaptatif, il faut mettre en place une solution capable d'utiliser en temps réel le modèle usager pour modifier le système selon les besoins de ce dernier. De plus, cette modification doit être efficace dans le sens où elle doit permettre l'atteinte de l'objectif fixée qui peut être l'apprentissage, la

motivation, le plaisir de jouer, etc. Le modèle de l'usager en lui-même ne sert qu'à modéliser l'usager d'un système mais pas à apporter des actions correctives dans ce système pour améliorer son apprentissage. C'est dans ce cadre que s'insère un moteur d'adaptation dont le but principal est d'interroger le modèle de l'usager de façon continue, et selon l'état actuel observé, proposer des actions correctives (règles pédagogiques) permettant de l'amener vers un état qu'on estime idéal.

4.4.1 Moteur d'adaptation

Le moteur d'adaptation que nous proposons est basé sur le mécanisme d'agent intelligent en intelligence artificielle. Un agent intelligent est une entité qui peut percevoir son environnement à l'aide de ses capteurs et agir dans son environnement à l'aide de ses effecteurs. Les capteurs de notre moteur d'adaptation représenteront les modèles modélisant l'apprenant, et les effecteurs représenteront les mécanismes lui permettant d'appliquer une règle d'adaptation. L'intelligence de notre moteur d'adaptation peut être vue comme une fonction (règles d'adaptation) qui associe un historique de données sensorielles (état de connaissances de l'apprenant joueur par exemple) à une action (partie droite des règles d'adaptation). La fonction objective étant l'amélioration des connaissances de l'apprenant joueur. Notre moteur

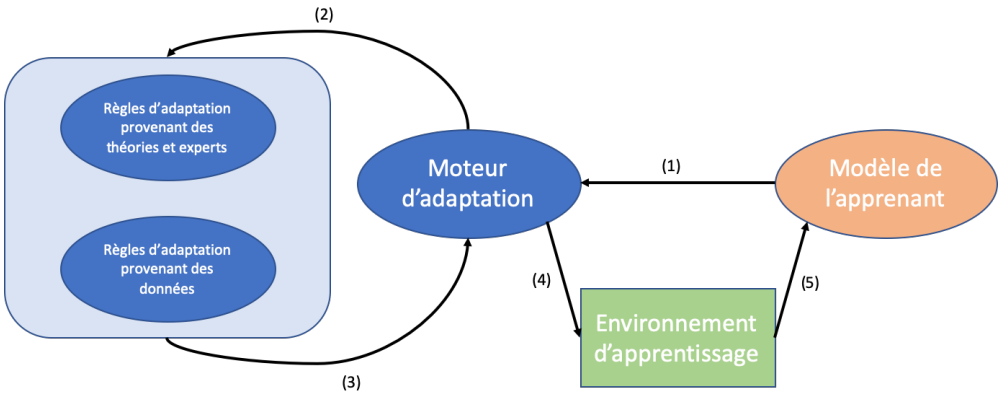


Figure 4.9 Moteur d'adaptation proposé.

d'adaptation sera donc implémenté comme un agent (voir figure 4.9) qui perçoit son environnement ((1) interroge le modèle de l'apprenant), sélectionne l'action à effectuer ((2 et 3) choisit la règle d'adaptation qui sera utilisée selon ce qu'il a observé) et agit sur l'environnement ((4) exécute la règle d'adaptation) et observe de nouveau l'environnement une fois l'environnement mis à jour selon la règle précédente (5). Cette boucle sera effectuée jusqu'à ce que l'objectif soit atteint : l'état des connaissances de l'apprenant soit satisfaisant (dans le cas d'un système tutoriel intelligent). Le résultat de cette phase nous permet à la fois d'évaluer le moteur d'adaptation en lui-même et le modèle de l'apprenant (s'il est capable de modéliser avec précision l'apprenant).

Comme nous pouvons le voir, le coeur de ce modèle repose sur les règles d'adaptation. Une règle d'adaptation doit être définie avec pour objectif l'amélioration de l'apprentissage, de la motivation, le l'expérience de jeu, etc. Comment définir ces règles et à quels moments les activer ? Une règle d'adaptation se présente généralement sous la forme $SI\ X\ ALORS\ Y1\ SINON\ Y2$ où X représente les évènements observés (actions posées par l'apprenant, l'état actuel de ses connaissances, etc.) et Y_1,Y_2 représente les actions à effectuer par rapport à l'observation faite (changement de musique, choix du prochain exercice, etc.). Pour définir les règles d'adaptation nous proposons de le faire de deux façons :

- Une partie des règles sera extraite des théories et ou des experts ;
- Une autre partie sera extraite d'algorithmes d'apprentissage machine telle que les arbres de décision et les réseaux de neurones.

L'ensemble des règles ainsi définies serviront au moteur d'adaptation pour l'adaptation en temps réel.

4.4.2 Extraction des règles à partir des théories

Ici les règles proviendront à la fois des théories et des experts du domaine d'apprentissage visée. Des exemples de règles peuvent être ceux liés aux styles d'apprentissage. Ces styles d'apprentissage sont bien connus ainsi que les caractéristiques associées à chacun. Il existe dans la littérature plusieurs solutions proposant des règles d'adaptation pour chacun de ces styles d'apprentissage (adaptation de l'interface, des éléments d'apprentissage (image, texte, etc.), séquencement des problèmes) (Karagiannidis et Sampson, 2004; Popescu *et al.*, 2010).

Voici les questions que nous avons identifiées et dont les réponses permettent d'identifier et d'extraire les règles d'apprentissage efficaces :

- Quelles sont les différentes connaissances en jeu ?
- Quels sont les prérequis pour mieux assimiler chacune de ces connaissances ?
- Quels sont les éléments (émotions, décor de l'environnement, affichage de l'état des connaissances, etc.) qui peuvent perturber l'apprentissage de ces connaissances ?
- Quels sont les éléments qui peuvent améliorer l'apprentissage de ces connaissances ?
- Quelles sont les conditions idéales pour un apprentissage efficace en général ?
- Quelles sont les règles d'interventions (interventions du système auprès du joueur-apprenant). En quelque sorte quelles sont les règles tutorielles ?

Selon les domaines, il peut être facile d'avoir accès aux règles d'adaptation théoriques ou pas. Cette manière d'obtenir les règles d'adaptation est la plus répandue dans les systèmes d'apprentissage (Ruiz *et al.*, 2008). Les règles identifiées manuellement peuvent être complétées avec celles identifiées automatiquement. Nous proposons des techniques qui permettent d'extraire d'autres règles en complément

à celles définies manuellement.

4.4.3 Extraction automatique des règles d'adaptation à partir d'arbres de décision et des réseaux de neurones

Le but est de déterminer automatiquement les attributs des apprenants et les éléments du système qui ont un impact sur l'apprentissage. Une fois cette étape complétée, il faut déterminer automatiquement les actions à prendre et les éléments du système à modifier lorsque certaines valeurs pour ces attributs sont observées. Ces deux étapes constituent le processus à suivre pour l'écriture des règles d'adaptation. Nous proposons d'extraire les règles d'adaptation directement à partir des données, en utilisant les arbres de décision et les réseaux de neurones.

4.4.3.1 Extraire des règles à l'aire d'un arbre de décision

Un arbre de décision est un outil de support à la décision ayant une structure sous forme d'arbre qui est apprise automatiquement à partir des données. C'est une technique d'apprentissage machine supervisée dont le résultat est interprétable en autant que la taille de l'arbre soit raisonnable, comparé à la plupart des autres techniques. La procédure de génération d'un arbre de décision à partir d'un ensemble d'apprentissage est appelée «induction de l'arbre». L'arbre généré permet une extraction facile des règles qui se lisent de la racine aux feuilles. Il y a deux types de noeuds dans l'arbre : les nœuds internes et les noeuds terminaux. Le noeud interne se déploie en différentes branches selon les différentes valeurs que l'attribut peut prendre (Ex : valence des émotions > 0 ou < 0). Le noeud terminal quant à lui décide la classe assignée (Ex : échec ou réussite à une connaissance).

Afin de générer les règles d'adaptation, notre solution consiste à construire un arbre de décision à partir des données des apprenants dont la classe à prédire est la performance sur les connaissances en jeu. L'idée étant de construire un

arbre de décision pour chaque connaissance c'est-à-dire, trouver les attributs et leurs valeurs qui agissent sur l'apprentissage de la connaissance. L'inconvénient d'une telle solution est que si le nombre de connaissances et très grand il devient moins efficace de construire autant d'arbres qu'il y a d'éléments de connaissances. Il faut également garder en tête que les règles ainsi extraites ne dépendent que des données utilisées pour l'apprentissage du modèle, et donc peuvent ne pas être généralisables.

4.4.3.2 Extraire des règles à partir d'un réseau de neurones

De nombreuses approches ont été proposées pour l'extraction des règles à partir des réseaux de neurones. Essentiellement, les algorithmes d'extraction de règles appartiennent à trois catégories : décompositionnelle, pédagogique et électif (Bologna et Hayashi, 2018). Notre proposition se situe dans les techniques de décomposition, où les règles sont extraites au niveau des neurones cachés et des neurones de sortie en analysant les valeurs de poids (Murdoch et Szlam, 2017; Lu *et al.*, 2017). À notre connaissance, aucune approche existante ne xtraite de l'extraction de règles à partir d'un réseau neuronal à des fins d'adaptation dans des systèmes d'apprentissage. Les règles que nous voulons extraire du réseau de neurones sont cachées à la fois dans sa structure et dans les poids attribués aux liens entre les noeuds.

Cependant, pour que les règles extraites des réseaux de neurones aient du sens, il faudra mettre des contraintes aux réseaux puisque ces structures sont des boîtes noires à la base. De plus, l'apprentissage est non déterministe ce qui veut dire que chaque nouvel apprentissage entraîne un nouveau résultat. Rappelons qu'un réseau de neurones implémente une fonction qui prend des données en entrées et produit des prédictions. Chaque entrée a une certaine importance dans la prédiction de la sortie. Cette importance est mesurée par le poids qu'attribue le réseau à chacune des entrées. Par exemple, si on considère un RN à une seule couche, avec 2 neurones

sur la couche d'entrées, alors la sortie s'écrirait :

$$Y = f(w_1 \cdot x_1 + w_2 \cdot x_2) + b, \tag{4.10}$$

où f est une fonction d'activation, b est le biais et les w_i sont les poids appris par le réseau et qui mesurent l'importance de chacune des entrées sur la prédiction de la sortie. Par exemple si $w_1 > w_2$ alors cela implique que l'entrée x_1 a plus d'importance dans le calcul de y que x_2. Ceci n'est vrai que si on contraint le réseau à apprendre uniquement des poids positifs et que l'on transforme les entrées de façon à ce que leur domaine de définition soit le même. Il a été prouvé qu'en contraignant ainsi la valeur des poids, on n'atténue pas le processus d'apprentissage du réseau et les résultats (Chorowski et Zurada, 2015). De notre exemple on peut extraire une règle qui dit que si x_1 est grand alors Y l'est aussi, sachant que f est une fonction d'activation strictement croissante sur $\mathbb{R}$. Cependant, si le réseau devient plus grand (plusieurs couches), il devient difficile d'évaluer cette importance et donc d'en extraire des règles. Ainsi, les règles extraites à partir du réseau de neurones seront comparées ou jumelées à celles extraites à partir de l'arbre de décision.

4.5 Conclusion

Dans ce chapitre, nous avons présenté nos solutions de modélisation de l'usager et d'adaptation. Le chapitre suivant sera dédié à l'utilisation et la validation de ces solutions dans des cadres applicatifs réels.

CHAPITRE V

APPLICATIONS DES SOLUTIONS ET RÉSULTATS

Afin d'évaluer les différentes solutions présentées au chapitre précédent, ainsi que leur efficacité dans au contexte éducatif, nous avons choisi comme cadre applicatif un jeu sérieux (*LesDilemmes*) et un STI (*Muse-logique*). Ces deux systèmes concernent l'apprentissage du raisonnement. Le raisonnement est fondamentalement un processus cognitif visant à tirer une information appelée conclusion à partir des informations données ou observées appelées prémisses par l'application d'une règle (Evans *et al.*, 1993). Le premier système vise l'apprentissage du raisonnement sociomoral alors que le second système vise l'apprentissage du raisonnement logique.

Dans ce chapitre, nous décrivons dans un premier temps nos deux systèmes ainsi que la façon dont nos modèles y ont été intégrés et utilisés. Ensuite, des expérimentations visant l'évaluation ainsi que les résultats des solutions sont présentés. Certaines des solutions proposées — en particulier le modèle multimodal — n'ont pas pu être testées dans ces deux systèmes. Néanmoins, nous les avons testées dans le cadre d'autres projets de recherche que nous présentons à la fin de ce chapitre.

5.1 LesDilemmes : un jeu pour l'amélioration du raisonnement sociomoral

Comme indiqué au chapitre 2, LesDilemmes est un jeu sérieux développé conjointement par l'Université du Québec à Montréal et l'Université de Montréal, dont l'objectif est d'évaluer et d'améliorer les compétences en raisonnement sociomoral du joueur-apprenant. La mise en marche de cet environnement a nécessité :

- L'intégration d'un modèle permettant l'évaluation automatique du niveau de raisonnement sociomoral du joueur en fonction de ses justifications verbales ;
- L'intégration d'un moteur d'adaptation éditable ;
- La mise en place des éléments d'adaptation dont la musique, les mouvements non-verbaux des personnages non joueurs (PNJs), etc.

Le jeu comporte 9 dilemmes qui sont :

- **Dilemme 1 (Petite soeur)** : Le joueur doit décider entre aller jouer avec ses amis et s'occuper de sa petite soeur comme demandé par sa mère, sachant que ses amis sont là pour aller jouer avec lui comme ils avaient prévu ;
- **Dilemme 2 (Le portefeuille)** : Le joueur doit décider de garder ou remettre un portefeuille tombé au sol, sachant que le propriétaire l'a laissé tomber devant lui et ses amis ;
- **Dilemme 3 (Le feu de circulation)** : Le joueur doit décider s'il traverse ou pas une route lorsque le feu de la circulation est rouge, sachant qu'il n'y a pas de voitures qui passent ;
- **Dilemme 4 (Se moquer de son ami)** : Le joueur doit décider s'il se moque ou pas d'un de ses amis qui est tombé, sachant que les autres sont en train de rire ;
- **Dilemme 5 (Tricher à l'examen)** : Le joueur voit un de se ses amis

tricher pendant un examen et doit décider s'il le dira à la professeure ou pas ;

- **Dilemme 6 (Voler des chocolats à l'épicerie)** : Le joueur doit décider s'il prend ou pas les chocolats en cachette sur l'étalage d'une épicerie sachant qu'il aime les chocolats et que le vendeur ne regarde pas ;
- **Dilemme 7 (Vitre cassée)** : Le joueur a cassé la vitre d'une voiture par accident en jouant avec ses amis et doit décider s'il en fait part au propriétaire du véhicule ou pas sachant que le propriétaire n'est pas sur les lieux ;
- **Dilemme 8 (Lancer des roches sur un chien**) : Le joueur assiste à une scène ou ses amis sont entrain de jeter des pierres à un chien, il doit décider s'il doit jeter les pierres comme ses amis ou pas ;
- **Dilemme 9 (Mauvaise note)** : Le joueur a obtenu une mauvaise note (F) à un examen et doit décider s'il l'a change pour un A ou pas avant de montrer à ses parents.

L'ordre respectif de difficulté croissante de ces dilemmes est : 8,9,7,3,4,1,5,2,6. Cet ordre est déterminé à partir de la première expérimentation du jeu. En général, les dilemmes les plus difficiles sont à la fin et les plus faciles sont au début du jeu.

Le modèle de l'apprenant dans le jeu comporte deux parties : un modèle permettant de prédire son niveau de raisonnement sociomoral sachant son justificatif verbal, et un modèle représentant son état émotionnel. Le modèle global a été implémenté comme un objet indépendant qui est mis à jour au fur et à mesure des interactions avec le système et qui est consulté continuellement par le moteur d'adaptation. Ici les émotions du joueur sont représentées par une mémoire déclarative alors que les connaissances sont modélisées par notre architecture hybride. Certaines règles d'adaptation ont été extraites à partir d'un arbre de décision et un réseau de neurones et d'autres ont été fournies par les experts. Nous avons

développé sous *Unity*, un outil permettant de spécifier facilement les règles d'adaptation (provenant d'experts et d'apprentissage machine) qui sont utilisées dans le jeu. Cet outil met en relation les attributs observables de l'usager et les éléments du jeu que l'on peut modifier. Les figures 5.1 et 5.2 présentent l'interface d'édition des règles d'adaptation sous *Unity*. Chaque règle définie dans cette interface est directement prise en compte dans le jeu.

Figure 5.1 Éditeur de règles sous *Unity* : Règles pour la boucle interne (*Inner loop*).

Nous avons, dans un premier temps, évalué les performances du modèle du joueur permettant de prédire son niveau de raisonnement. Nous avons par la suite intégré ce modèle dans le jeu, ainsi que les règles d'adaptation mises en place. Une première expérience a été conduite avec une version du jeu ne contenant pas les règles d'adaptation. Cette expérience s'est effectuée sur 30 participants âgés de 8 à 19 ans. Une deuxième expérience a ensuite été faite avec la version adaptative du jeu, dans laquelle nous avons intégré le modèle de l'apprenant ainsi que les règles d'adaptation. 40 participants âgés de 10 à 17 ans y ont participé. Nous avons me-

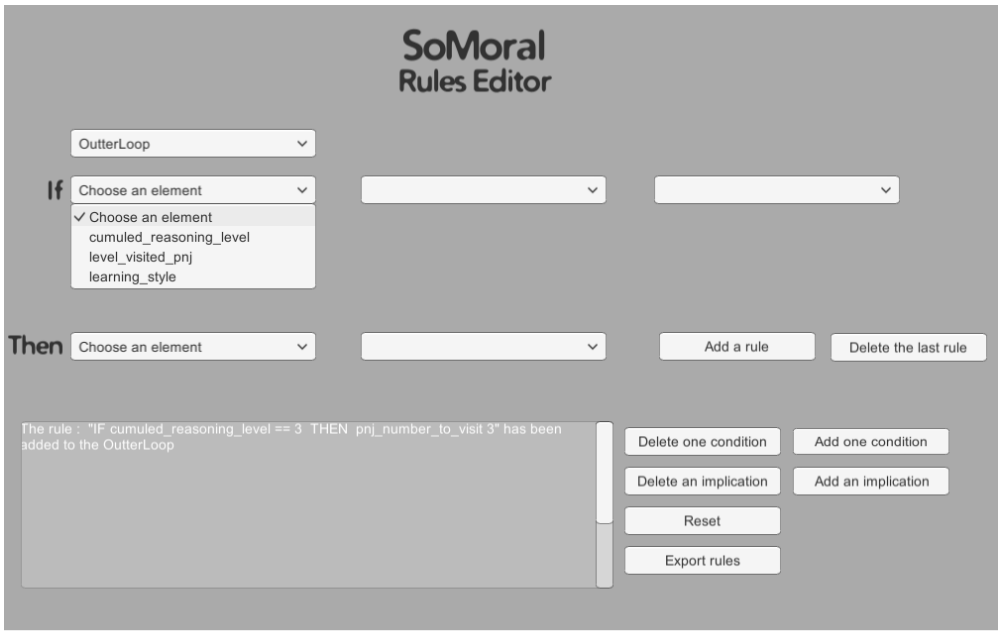


Figure 5.2 Éditeur de règles sous *Unity* : Règles pour la bouucle externe (*Outter loop*).

suré la différence entre ces deux systèmes tant au niveau du gain d'apprentissage, de l'immersion que de la satisfaction. De plus nous avons évalué l'évolution des connaissances des apprenants, avant et après le jeu dans sa version adaptative, afin d'évaluer le gain d'apprentissage net attribuable à la version adaptative.

5.1.0.1 Conception du modèle usager

Le modèle du joueur-apprenant mis en œuvre dans l'environnement d'apprentissage comporte deux dimensions clé : l'état affectif et le profil cognitif qui implémente principalement le raisonnement sociomoral qui est la connaissance visée. Le modèle usager dans le jeu comprend un ensemble d'attributs dont les valeurs sont mises à jour au fur et à mesure des interactions avec le système, et un modèle d'apprentissage machine pour la prédiction du niveau de raisonnement sociomoral. Les attributs de l'usager considérés ici sont :

1. le niveau de raisonnement cumulé du joueur ;

2. son niveau de raisonnement en progrès (vrai ou faux) calculé après chaque dilemme ;
3. son style d'apprentissage : conformiste, résistant, intentionnel, ou performant) — qui n'est pas encore utilisé — ;
4. son style de visite des PNJ : tous ou en partie, visite plutôt les forts (4 et 5) versus les faibles (1 à 3) ;
5. son score (nombre d'amis + nombre de *like*/*dislike*) ;
6. ses émotions courantes : 7 émotions à partir de Facereader ainsi que la valence et l'arousal ;
7. son niveau de raisonnement courant : valeur numérique de 0 à 5 ;
8. l'évaluation des PNJ : en accord/désaccord ;
9. ses décisions : réponses aux questions posées par rapport aux dilemmes et qui précèdent ses justificatifs.

Le score dans le jeu est constitué du nombre de *like/dislike* qui se mettent à jour en fonction du niveau de raisonnement du joueur et le nombre d'amis qui se met à jour en fonction de l'évaluation que fait le joueur sur les opinions des autres (en accord/désaccord). Le calcul du nombre de *like/dislike* est fait de la manière suivante :

- Si le joueur donne un justificatif de niveau 1, alors son nombre de *likes* augmente de 5 et son nombre de *dislikes* augmente de 5 également. (On évite ainsi de donner un retour négatif à l'apprenant) ; (voir l'annexe F.1)
- Si le joueur donne un justificatif de niveau 2, alors son nombre de *likes* augmente de 6 et son nombre de *dislikes* augmente de 2 ;
- Si le joueur donne un justificatif de niveau 3, alors son nombre de *likes* augmente de 7 et son nombre de *dislikes* augmente de 1 ;
- Si le joueur donne un justificatif de niveau 4, alors son nombre de *likes* augmente de 8 et son nombre de *dislikes* augmente de 0 ;

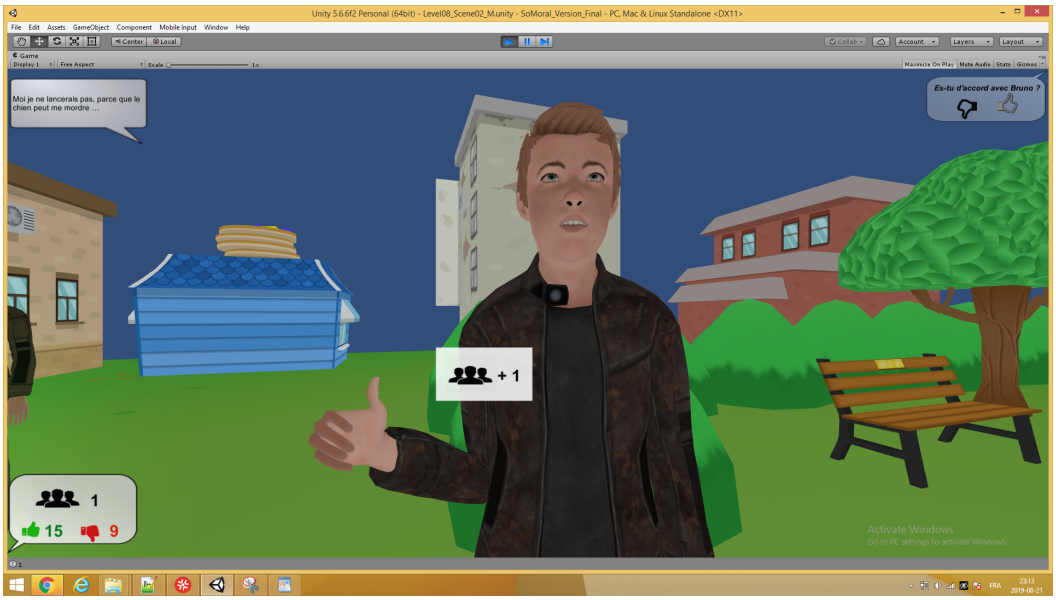


Figure 5.3 Réaction positive d'un PNJ lorsque le joueur l'a bien évalué : le niveau du joueur $\leq$ à celui du PNJ et le joueur est en accord avec l'opinion du PNJ ; ou le niveau de raisonnement du joueur $>$ au niveau du PNJ et il est en désaccord avec l'opinion du PNJ.

- Si le joueur donne un justificatif de niveau 5, alors son nombre de *likes* augmente de 9 et son nombre de *dislikes* augmente de 0.

Concernant le nombre d'amis la mise à jour est faite de la manière suivante :

- Si l'apprenant est de niveau inférieur ou égale au niveau du PNJ qu'il évalue et que l'évaluation est positive (en accord) alors il gagne un nouvel ami (ex : voir figure 5.3) ;
- Si l'apprenant est de niveau supérieur au niveau du PNJ qu'il évalue et que l'évaluation est négative, alors il gagne un nouvel ami ;
- Dans tous les autres cas, il ne gagne pas de nouvel ami.

Il est à noter que ces calculs sont faits sur une base intuitive, l'idée étant de ne pas pénaliser le joueur mais de moins le récompenser si son niveau de raisonnement sociomoral est moins élevé. La figure F.3 en annexe présente un extrait du modèle usager tel qu' implémenté dans le jeu LesDilemmes.

Comme indiqué dans (Birk *et al.*, 2015), un modèle usager qui peut représenter en long et en large avec précision cet usager dans un système, conduit à une adaptation efficace, ce qui peut aider à augmenter la satisfaction et la motivation du joueur. Pour ce faire, il est important de s'assurer de l'efficacité du modèle de l'usager avant de déployer le système à des fins réelles. Puisque l'état affectif de l'apprenant est représenté par une mémoire déclarative (les états émotionnels sont stockés sans autre traitement), nous allons uniquement nous concentrer sur la validation du profil cognitif. Calculer le niveau de raisonnement sociomoral d'un individu nécessite un examen des justificatifs verbaux qu'il fournit pour résoudre les dilemmes durant le jeu. Ceci implique l'implémentation d'un modèle capable de mesurer automatiquement cette donnée pendant le jeu. Nous disposons pour ce faire d'un ensemble de données textuelles déjà annotées par des experts et provenant de l'expérimentation du test SoMoral original. Ces données annotées sont accompagnées d'une description associée (un paragraphe avec des concepts-clés) associée à chaque niveau (ou classe) de maturité sociomoral. Nous avons développé un modèle d'apprentissage automatique hybride qui permet d'évaluer avec précision le niveau de compétence en raisonnement sociomoral d'un joueur à partir d'un verbatim (justificatif) émis.

Les descriptions de chaque niveau de raisonnement sont utilisées comme étant des connaissances a priori pour notre modèle hybride. Nous avons utilisé deux techniques différentes permettant de rendre ces connaissances utilisables par le modèle hybride. La première technique est le *Word Movers' Distance* (WMD) (Kusner *et al.*, 2015) et la deuxième est les n-grames (Damashek, 1995). Le WMD est une technique qui permet de soumettre une requête et de renvoyer les documents les plus pertinents à la requête. Elle permet d'évaluer de manière significative la «distance» entre deux documents, même lorsqu'ils n'ont pas de mots en commun. L'hypothèse est que des mots similaires devraient être liés à des

vecteurs similaires. Au lieu d'utiliser la distance euclidienne et d'autres mesures de distance basées sur les *bags-of-words*, ils ont proposé d'utiliser le *word2vec* pour calculer les similarités. Le but d'utiliser ces techniques est de pouvoir récupérer la donnée en entrée et de la comparer avec la connaissance en utilisant ces solutions et d'en sortir avec un résultat ; résultat qui sera donc jumelé avec celui extrait de l'architecture neuronale par mécanisme d'attention.

Nous avons utilisé le WMD pour construire la première partie du contenu du vecteur représentant les connaissances dans notre modèle hybride. Chaque description de chaque niveau de raisonnement sociomoral ainsi que chaque verbatim sont considérés comme des «documents» différents. Nous avons transformé chaque document en une représentation utilisant les modèles *word2vec* pré-entraînés pour le français [1] [2]. Nous avons calculé la similitude entre chaque verbatim et chaque description (5 descriptions) en utilisant l'outil Gensim *Wmd-Similarity* [3]. Pour chaque verbatim, ce calcul nous a fourni un vecteur de longueur 5 où chaque entrée est la similitude entre le verbatim et la description du niveau de compétence en raisonnement sociomoral correspondant. Les valeurs se situent entre 0 et 1. Nous avons extrait les n-grammes (uni et bi-grammes) de la description textuelle des niveaux, ce qui nous a donné une liste de n-grammes pour chaque niveau de raisonnement. Nous avons également généré une liste de synonymes de tous les mots-clés (extraits manuellement) inclus dans les descriptions. Pour chaque verbatim, nous avons compté le nombre de fois que chaque n-gramme de chaque niveau apparaissait dans le texte. La somme des vecteurs générés par ce processus nous a donné, pour chaque verbatim, un vecteur de taille 5 où chaque entrée

1. http ://fauconnier.github.io/
2. http ://wacky.sslmit.unibo.it/doku.php ?id=corpora
3. https ://tedboy.github.io/nlps/generated/generated/gensim.similarities.WmdSimilarity.html

représente le nombre de n-grammes et le nombre de synonymes de chaque niveau trouvé dans le verbatim. Nous avons finalement appliqué la fonction *softmax* au vecteur résultant qui a été ajouté à celui généré par la méthode WMD. Le vecteur résultat v de cette phase est le vecteur «expert» de notre modèle hybride.

Dans l'équation 4.4 qui permet de calculer la prédiction à partir du modèle hybride, le e_k ($k = 5$) est le niveau de raisonnement sociomoral actuel (normalisé, somme de tous les éléments = 1) calculé à partir des connaissances a priori (vecteur v ci-dessus). La valeur e_k est un vecteur de longueur égal au nombre de niveaux (5 dans notre cas) et chaque entrée représente la probabilité prédite par les connaissances a priori que le verbatim appartient à chacune des classes de raisonnement. Le code pour ce modèle est disponible publiquement sur Github [4].

5.1.0.2 Conception du moteur d'adaptation

Afin d'introduire une forme de rétroaction et d'assignation de score à l'intérieur de notre jeu, nous avons ajouté une rétroaction social simulée, montrant le nombre d'«amis» et de «*likes*» selon les réponses des joueurs. Quand le niveau de maturité du raisonnement sociomoral du joueur augmente, le joueur gagne des «*likes*», et quand il évalue positivement (en accord) les opinions des PNJ ayant un niveau de maturité plus élevé que le sien, il se fait des «amis». Par contre, s'il est en accord avec des PNJs ayant des niveaux de maturité plus bas, il n'a aucun effet sur le nombre de «*likes*» ou le nombre d'«amis». Ces règles, qui découlent d'une interaction avec des experts en psychologie du raisonnement sociomoral, sont détaillées en annexe D.

La mise en place de l'adaptation a consisté à :

- Identifier les items modifiables du jeu (gestuelles des PNJs, ordre des di-

4. https ://github.com/angetato/Combining-DNN-With-Expert-Knowledge-ToPredict-sociomoral-Reasoning-Skills

lemmes, musiques et messages) et les attributs du modèle du joueur auxquels ces items sont sensibles (émotions, niveaux de raisonnement, score).

- Mettre en place des règles d'adaptation (que nous présentons ci-dessous) extraits à partir d'experts, de théories et des données.
- Développer un éditeur de règles d'adaptation sous *Unity*.

Nous avons opté pour 2 stratégies de personnalisation qui sont la boucle externe (*Outer Loop*) et la boucle interne (*Inner Loop*) (Vanlehn, 2006). La boucle externe vise la construction du dilemme (choix de la prochaine activité) alors que la boucle interne vise le déroulement du dilemme (choix des modalités pendant que l'apprenant résout le dilemme). Pour chaque stratégie, nous avons identifié les attributs du modèle de l'usager qui apparaîtront dans la partie «SI» des règles d'adaptation. Les attributs de l'usager dont le système a besoin de surveiller les valeurs pour l'exécution de la *Outer Loop* sont : (1) son niveau de raisonnement cumulé, (2) son niveau de raisonnement en progrès (vrai ou faux) calculé après chaque dilemme, (3) son style d'apprentissage (4 valeurs possibles — conformiste, résistant, intentionnel, performant—), (5) son style de visite des PNJ et (6) son score (nombre d'amis + nombre de *like*/*dislike*). Dans le cas de la boucle interne, ce sont plutôt (1) ses émotions courantes, (2) son niveau de raisonnement courant (valeur numérique de 0 à 5), (3) l'évaluation observée que fait le joueur des PNJ (en accord/désaccord) qui sont observés et (4) ses décisions qui sont prises en compte.

Nous avons identifié pour chaque stratégie de personnalisation, les artefacts du jeu pouvant être modifiables pour des fins d'adaptation. Pour la construction du dilemme nous avons :

- Mode de calcul du score (en fonction du style d'apprentissage) ;
- Difficulté du jeu (ordre des dilemmes) : dilemmes ordonnés en fonction des émotions négatives/ positives suscitées lors de la première expérimentation,

dilemmes les plus faciles vers les plus difficiles ;

- Obligation de consulter tous les PNJs ou pas selon le style de consultation du joueur ;
- Messages d'encouragements en fonction de l'évolution du niveau de raisonnement
- Musique / volume, choix, musique de fond.

Pour ce qui est du déroulement du dilemme (*inner loop*) nous avons :

- Musique / volume, choix, musique de fond ;
- Expressions non verbales des PNJ en fonction de l'évaluation du joueur (les PNJ peuvent réagir à l'évaluation de la réponse du joueur avec le pouce vers le haut ou vers le bas et avec une expression faciale) ;
- Rétroaction sur le niveau de raisonnement courant (messages d'apprentissage, motivation, etc.) : encourager quand le niveau de raisonnement monte, etc.

Ainsi notre méta-modèle des règles d'adaptation est défini comme suit. Les règles de construction du dilemme sont définies suivant le patron :

- **Partie SI** : Attributs du modèle de l'apprenant pour la construction du dilemme (choix du prochain dilemme, choix du nombre de PNJs à visiter, etc.), informations sur le dilemme précédent (style de visite des PNJs, etc.) ;
- **Partie ALORS** : Artefacts modifiables pour la construction du dilemme.

Les règles de déroulement du dilemme sont définies suivant le patron :

- **Partie SI** : Attributs du modèle de l'apprenant pour la construction du dilemme et attributs du modèle de l'apprenant pour le déroulement du dilemme ;
- **Partie ALORS** : Artefacts modifiables pour le déroulement du dilemme.

À ces règles, viendront s'ajouter celles apprises par arbre de décision et réseaux de neurones. Les figures 5.4 et 5.5 résument le méta-modèle des règles d'adaptation. Ce méta-modèle est implémenté entièrement dans notre outil de définition de

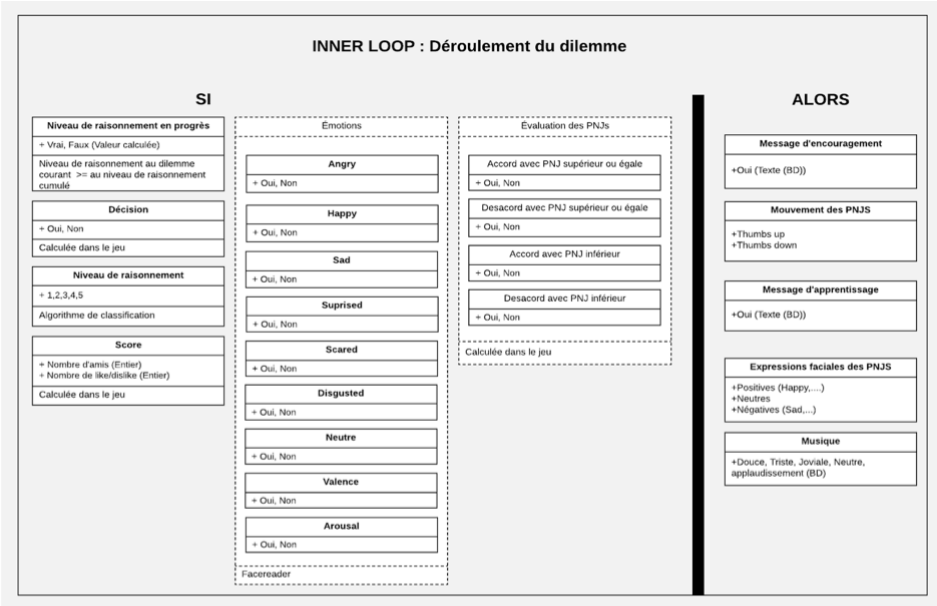


Figure 5.4 Méta-modèle pour la construction des règles d'adaptation pour le déroulement du dilemme.

règles sous *Unity* (voir figures 5.1 et 5.2).

Toutes les règles d'adaptation actuellement implémentées dans le jeu sont créées à partir de ce méta-modèle (via notre outil). Certaines des règles ont été extraites automatiquement des données et les autres à partir des théories et des experts. L'ensemble des règles est disponible en annexe (annexe D) ainsi que les différents messages (encouragements, félicitations, apprentissage, etc.) fournis par les experts (voir annexe C). Nous présentons quelques règles d'adaptation et quelques messages.

1. Règles définies par les experts et les théories :
   - Si le joueur de niveau de raisonnement R n'est pas d'accord avec un PNJ de niveau de raisonnement R_x et que $R_x \geq R$ **Alors** le PNJ affichera une émotion négative et orientera son pouce vers le bas pour indiquer au joueur qu'il ne devrait pas être en désaccord.

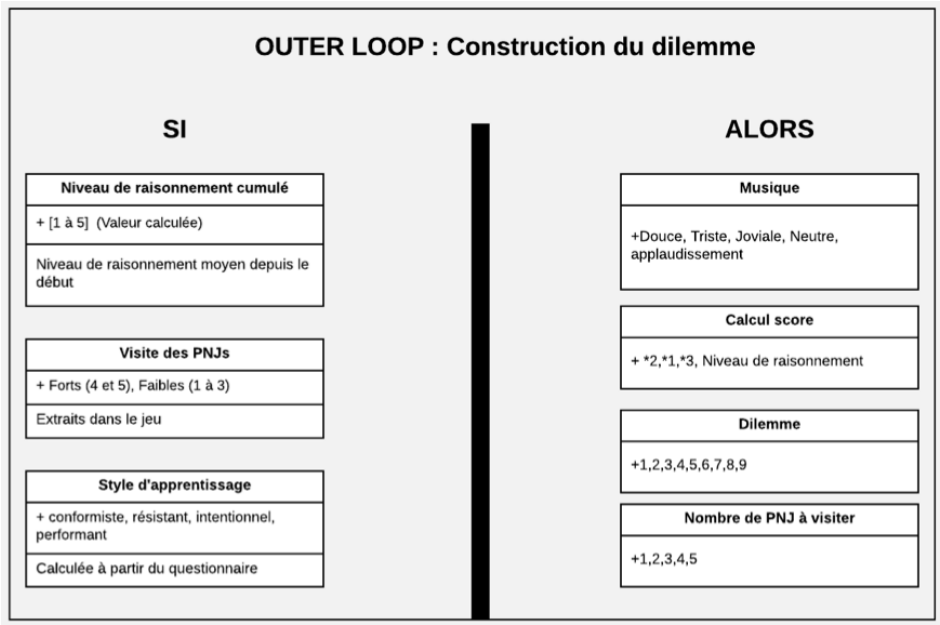


Figure 5.5 Méta-modèle pour la construction des règles d'adaptation pour la construction du dilemme.

- **Si** le niveau de raisonnement du joueur est en progrès **Alors** messages de félicitations sur le niveau actuel.
- **Si** le joueur est de niveau : **1** (Peur de l'autorité) **Alors** message qui l'aidera à penser comme quelqu'un de niveau 2 ou 3 (le choix du message est automatique dans le jeu) (voir l'annexe F.2). **2** (égoiste) **Alors** message qui l'aidera à penser comme quelqu'un de niveau 3 ou 4. **3** (relations interpersonnelles) **Alors** message qui l'aidera à penser comme quelqu'un de niveau 4 ou 5.

2. Règles définies par apprentissage machine : règles extraites de l'arbre de décision (figure 5.6) et du réseau de neurones (figure 5.7), modèles entraînés à partir des données récoltées pendant la première expérimentation sur le jeu non adaptatif. Chaque couche du réseau de neurones appris est déterminée par une matrice de poids et un vecteur de valeurs de biais. Nous présentons en annexe (E) des exemples de poids de chaque couche, tirés

de 3 entraînements différents. L'entraînement a atteint une précision de 65 ~70 % ce qui est assez raisonnable étant donné la quantité de données que nous avions et la simplicité de l'architecture utilisée, mais loin d'être parfait. C'est pourquoi les résultats finaux que nous présentons ci-dessous sont combinés avec les résultats de l'arbre de décision. De notre arbre de décision, nous pouvons extraire les observations suivantes :

- Un apprenant ayant une faible valeur d'*arousal* a plus de chance d'avoir un niveau de raisonnement supérieur à 3.
- Un apprenant qui est d'accord avec un PNJ de niveau 2 a plus de chance d'avoir un niveau de raisonnement supérieur à 3 s'il est moins dégoûté.
- Un apprenant qui n'est pas d'accord avec un PNJ de niveau 2 a plus de chance d'avoir un niveau de raisonnement inférieur à 3 s'il est moins surpris.
- Un apprenant qui n'est pas d'accord avec un PNJ de niveau 2 mais d'accord avec un PNJ de niveau 3 a plus de chances d'avoir un niveau de raisonnement supérieur à 3.

Ces observations sont basées sur la valeur de l'indice de *Gini* des attributs et du nombre d'échantillons respectant chacune de ces règles. Pour extraire les règles du RN, nous avons utilisé le processus suivant. Soit W la matrice de poids de l'avant-dernière couche où chaque rangée $W_j = [w_1, ..., w_n]$ représente les poids connectés à chaque neurone j de la dernière couche. $n = 3$ (nos 3 modalités) et $j \in$ 1,2 où $j = 1$ signifie que les poids sont reliés au neurone qui déclenche une valeur proche de 1 lorsque le niveau de raisonnement est inférieur à 3 ($[1, 0]$). Pour évaluer l'importance des entrées dans la prédiction de la sortie, nous n'avons pas directement considéré la matrice de poids, mais les valeurs relatives. Ainsi au lieu d'utiliser w_i nous avons utilisé a_i qui est le résultat de l'application de la fonction softmax

sur tous les poids connectés au même neurone.

$$a_{i,j} = \frac{e^{w_{i,j}}}{\sum_{i=1}^{n} w_{i,j}} \tag{5.1}$$

La valeur $A_{i,j}$ que nous avons utilisée pour évaluer l'importance de l'entrée i (sur la prédiction de chacune des sorties j) est calculée comme suit :

$$A_{i,j} = \frac{\sum_k a_{i,j,k}}{\sum_k \sum_j a_{i,j,k}} \tag{5.2}$$

Dans chaque couche, les entrées ayant une valeur supérieure à $A_{i,j}$ sont les plus importantes. Ce calcul est effectué pour toutes les couches et les résultats sont affichés dans les dernières colonnes des tableaux E.1,E.2,E.3 en annexe. Maintenant, si nous regardons la valeur A_i apprise par le RN, nous pouvons voir que :

- L'émotion (l'excitation, la surprise et la joie) est la caractéristique la plus importante par rapport aux autres caractéristiques pour prédire si le niveau de raisonnement est élevé ou non (voir les valeurs de $A_{i,1}$ et $A_{i,2}$ dans le tableau E.1).
- L'évaluation des PNJ de niveau de raisonnement 2 et 5 est plus importante que l'évaluation des autres PNJs (voir les valeurs de A_i dans le tableau E.2).

Sur la base de ces observations, nous pouvons voir que les caractéristiques les plus importantes pour prédire le niveau de raisonnement sont l'arousal, la surprise et les états émotionnels positifs (content, etc.), mais aussi l'évaluation faite sur les PNJs avec les niveaux de raisonnement 2 et 5. Les résultats sont assez semblables à ceux extraits à partir de l'arbre de décision. Cependant, nous ne pouvons pas spécifier les valeurs idéales pour ces caractéristiques à partir des poids appris, c'est pourquoi nous avons utilisé

l'arbre de décision comme support. L'un des avantages de l'utilisation du RN est que, lorsque les données deviendront de plus en plus volumineuses, le modèle sera toujours en mesure d'indiquer les caractéristiques à observer et à modifier pour améliorer la maîtrise des connaissances par rapport à l'arbre de décision. Les règles finales mises en œuvre dans LesDilemmes sont celles extraites du RN. De ces éléments, nous avons défini les règles suivantes :

- **Si** *Arousal* < 0.228 et Sad < 0.02 **Alors** *Musique* douce.
- **Si** le joueur n'a pas visité les PNJs de niveau 2 ou 5, **Alors** l'obliger à le faire dans les prochains dilemmes.

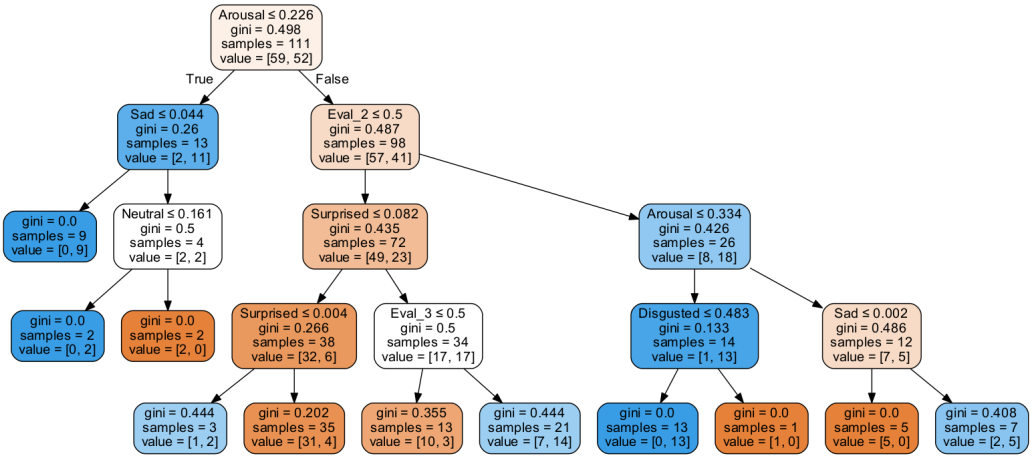


Figure 5.6 Arbre de décision appris à partir des données de la première expérimentation sur le jeu non adaptatif.

Quelques messages implémentés dans le jeu, dont la liste complète est disponible en annexe C :

Messages de félicitations :

- Bravo !
- C'est très bien.
- Félicitations !

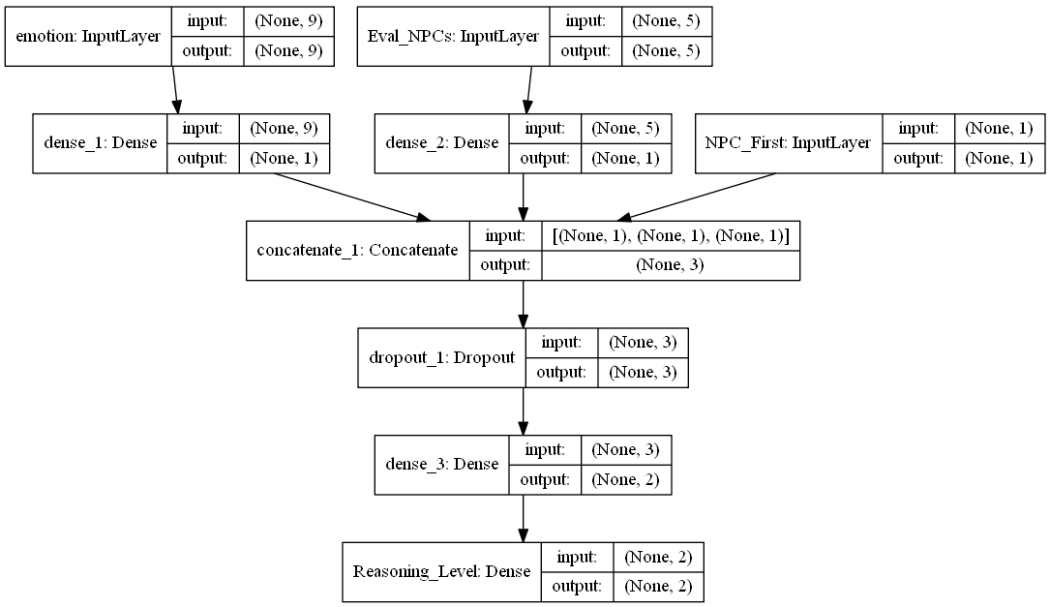


Figure 5.7 Le réseau de neurones multimodale proposé pour l'extraction des règles d'adaptation.

- Excellent !
- etc.

Messages d'apprentissage : Les messages d'apprentissage devraient suggérer le développement à venir, ce que l'apprenant devrait faire pour accéder au niveau de connaissance suivant (niveau de raisonnement supérieur).

1. **Messages pour score = 1 (Orientation vers la punition et l'obéissance à l'autorité) vers score = 2 :**
   - *Dilemme 1* : As-tu pensé que cela peut être avantageux pour toi de respecter l'entente que tu avais avec ta mère ?
   - *Dilemme 2* : Aimerais-tu qu'on te remette ton portefeuille si tu l'échappais ?
   - *Dilemme 3* : As-tu considéré que tu pourrais avoir un accident si tu traversais sur la lumière rouge ?
2. **Messages pour score = 2 (Orientation vers les échanges égocen-**

triques) vers score = 3 :

- *Dilemme 1* : As-tu pensé que ta petite soeur pourrait avoir besoin qu'on s'occupe d'elle ?
- *Dilemme 2* : As-tu considéré que la personne qui a échappé son portefeuille apprécierait qu'on lui redonne ?
- etc.

3. **Messages pour score = 3 (Orientation vers les relations interpersonnelles) vers score = 4** :
   - *Dilemme 1* : As-tu pensé que de respecter les ententes que l'on prend permet de conserver des relations harmonieuses ?
   - *Dilemme 2* : As-tu considéré que le portefeuille appartient à la personne et que l'on devrait respecter sa propriété en lui redonnant ?
   - etc.
4. **Score = 4 (Régulation de la société) et score = 5 (Évaluation du contrat social) : Messages d'apprentissage = Messages félicitations.**

Messages de Généralisation Les messages de généralisation sont des messages qui s'affichent lorsqu'un joueur(se) atteint un niveau de raisonnement pour la première fois. Ils visent à rappeler au joueur les spécificités du niveau qu'il vient d'atteindre.

- **Première fois le niveau 2 :** Yeah ! Tu as pensé aux avantages de la situation !
- **Première fois niveau 3 :** Bravo ! Tu as pensé à considérer les autres dans ta réponse au dilemme !
- etc.

5.1.0.3 Protocole expérimental

Comme nous l'avons déjà mentionné, le jeu a été évalué dans sa version non adaptative et sa version adaptative. La version adaptative contient le modèle de l'usager et le moteur d'adaptation alors que la version non adaptative n'en possède pas. Le jeu a été évalué sur des adolescents de 8 à 19 ans mais lors des évaluations nous avons limité l'étude aux jeunes de 10 à 17 ans inclusivement puisque les jeunes n'étant pas dans cette tranche n'avaient joué qu'à la version non adaptative, le but étant d'homogénéiser les deux groupes de participants. Les participants recrutés ont été récompensés financièrement. La version non adaptative a été testée en premier lieu sur 30 participants mais nous n'avons gardé que 23 à cause de l'âge et du bruit dans les données. Quant à la version adaptative, nous avons testé sur 47 participants mais n'avons gardé que 43 pour les mêmes raisons. Avant l'expérimentation, les joueurs sont amenés à remplir un questionnaire basé sur les styles d'apprentissage de Martinez dont le but était d'extraire leur style d'apprentissage. Ce questionnaire comportait également des questions sur leurs habitudes sur Internet (réseaux sociaux, etc). Le questionnaire complet est disponible à l'annexe B.1. Une fois le questionnaire rempli, les participants sont amenés à passer un pré-test sur ordinateur qui consiste en 3 dilemmes pré-sélectionnés par notre équipe. Pour chaque dilemme, le participant écoute le dilemme puis donne son opinion de manière verbale. Il n'y a eu aucune évaluation du système pendant cette phase, mais une évaluation manuelle a été faite par la suite pour l'évaluation. Le but du pré-test était d'évaluer le niveau de raisonnement de chaque participant avant le jeu, ce qui nous permettrait d'évaluer l'impact (au moins à court terme) du jeu sur le niveau de raisonnement avant/après. Une fois le pré-test terminé, le participant pouvait alors jouer au jeu. Pendant le déroulement du jeu, nous captions les émotions via les expressions faciales grâce à l'outil *Facereader*. Ces émotions étaient utilisées en temps réel dans la version adaptative du jeu. les données sur

le regard avaient aussi été récoltées avec la version non adaptative mais nous avons abandonné l'idée avec la version adaptative. La raison est que, l'analyse de ces données oculométriques ne nous avait pas donné d'informations significatives. Une fois le jeu terminé, les participants étaient alors amenés à répondre à un questionnaire final (voir annexe B.2) portant sur l'immersion, l'apprentissage et l'appréciation du jeu (efficacité/satisfaction). Une fois le questionnaire rempli, les participants effectuaient un post-test (similaire ou pré-test et avec des dilemmes qui se rapprochent) dont le but est d'évaluer le changement opéré par le jeu dans leur façon de raisonner socialement. Nous avons divisé l'évaluation des résultats d'expérimentation en deux parties :

- **Évaluation subjective** : Le but est d'évaluer comment le joueur perçoit le jeu tant au niveau de l'immersion, de l'apprentissage que de l'appréciation du jeu. Pour ce faire, nous utilisons principalement les réponses aux questionnaires initial et final.
- **Évaluation objective** : Le but est d'évaluer objectivement l'apprentissage et l'appréciation du jeu. L'apprentissage est évalué par l'analyse (experts et algorithme) des opinions non verbaux fournies par les participants pendant le pré-test, le jeu et le post-test. L'appréciation du jeu est évaluée par l'analyse des émotions ressenties pendant le déroulement du jeu.

5.1.1 Évaluation du modèle du joueur-apprenant

Les résultats présentés dans cette section ont fait l'objet de publications (Tato *et al.*, 2017b; Tato *et al.*, 2019a). Afin d'évaluer empiriquement le modèle hybride permettant la prédiction du niveau de raisonnement, nous avons étudié six modèles différents : un CNN (*cnn-only*), un LSTM (*lstm-only*), un CNN où les connaissances a priori ont simplement été concaténées aux caractéristiques apprises (*cnn-expert*), un LSTM où les connaissances a priori ont été concaténées

aux caractéristiques apprises (*lstm-expert*) et enfin les modèles proposés qui sont le *cnn-expert-att* (un CNN où les connaissances a priori ont été concaténées aux caractéristiques apprises en utilisant le mécanisme d'attention) et le *lstm-expert-att* (un LSTM où les connaissances a priori ont été concaténées aux caractéristiques apprises en utilisant le mécanisme d'attention). Nous avons utilisé la même initialisation des paramètres pour tous les modèles. AAdam a été utilisé comme algorithme d'optimisation avec un taux d'apprentissage réglé à 0.001 et un seuil $S = 0.001$. Les architectures ont été implémentées en utilisant Keras et les expériences ont été faites en utilisant les modèles Keras intégrés, ne faisant que de petites modifications aux paramètres par défaut.

5.1.1.1 Prétraitement des données

Les données se composent de 731 verbatims (en français) annotés manuellement par des experts. Les verbatims ne sont pas répartis également entre les classes (données déséquilibrées). Le tableau 5.1 montre la distribution des données où par exemple, les classes 4 et 5 ont un petit nombre de verbatims que les autres (1, 2 et 3). En fait, le niveau 5 étant le plus haut niveau de maturité, il est rare de rencontrer des jeunes et adolescents ayant ce niveau. Cela signifie que certaines de ces classes ont très peu d'exemples pour l'entraînement. Sur les 731 verbatims, 53 d'entre eux ont été classés 0, ce qui signifie que le verbatim ne représente pas un des niveaux de compétence du raisonnement sociomoral. Nous n'avons pas tenu compte de ces cas dans notre étude, ce qui a réduit notre corpus à 678 verbatims. Nous avons également limité notre problème de prédiction à 5 niveaux de raisonnement sociomoral : 1, 2, 3, 4 et 5. Les verbatims appartenant aux niveaux intermédiaires 1.5, 2.5, 3.5 et 4.5 ont été ajoutés aux verbatims appartenant respectivement aux niveaux 2, 3, 4 et 5. Le jeu de données à notre disposition est déséquilibré. Les premiers modèles que nous avons développés étaient biaisés vers des scores qui avaient plus d'échantillons. Par conséquent, nous avons appliqué

Tableau 5.1 Répartition des verbatims entre les différentes classes de raisonnement sociomoral.

Score	Fréquence	Score	Fréquence
1	232	1.5	11
2	76	2.5	29
3	207	3.5	31
4	40	4.5	3
5	49		

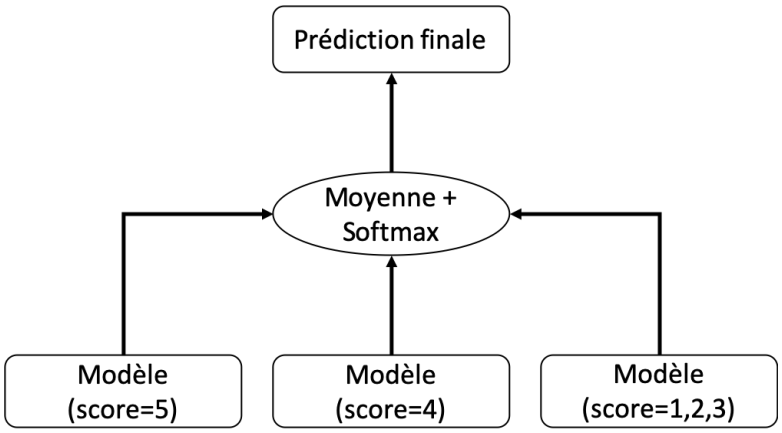


Figure 5.8 Modèle final pour la prédiction du niveau de raisonnement sociomoral. Chaque sous-modèle est spécialisé dans la prédiction du ou des niveaux spécifiés entre parenthèses. Chaque modèle est une architecture hybride.

certaines techniques visant à équilibrer l'ensemble de données. Nous avons entraîné différents modèles spécialisés dans la prédiction de chacun des niveaux de raisonnement sociomoral à problème (scores 4 et 5). Nous avons ensuite utilisé des méthodes d'ensemble pour combiner les résultats de ces modèles et produire la prédiction finale. Les méthodes d'ensemble sont des techniques qui créent plusieurs modèles et les combinent ensuite pour produire de meilleurs résultats (Dietterich, 2000) . Le vote et le calcul de la moyenne sont deux des méthodes d'ensemble les plus faciles à mettre en oeuvre. La stratégie de vote a été choisie, car elle convient bien aux problèmes de classification. L'architecture du modèle final est présentée à la figure 5.8.

5.1.1.2 Architectures pour la prédiction du niveau de raisonnement sociomoral

La figure 5.9 montre les 2 architectures que nous proposons pour la prédiction du niveau de raisonnement sociomoral. Les modèles prennent en entrée les verbatims qui ont été pré-traités (*tokenisation, text to sequence*, etc.) et vectorisés. Les vecteurs sont ensuite transmis à la couche de plongement (*embbeding*). Dans la figure 5.9 à gauche, les vecteurs issus de la couche de plongement sont passés à la couche LSTM. Il est à noter que nous n'avons considéré que la sortie de la dernière cellule. Les connaissances a priori et la sortie du LSTM sont ensuite transmises à la couche d'attention qui effectue une combinaison des deux sources de données (voir section 4.3.1). Le vecteur extrait de la couche d'attention est fusionné avec les connaissances a priori et la sortie du LSTM. La concaténation de ces trois sources de données est transmise à la dernière couche pour la prédiction finale. Ce processus est le même pour la CNN (voir figure 5.9 à droite), sauf que la couche d'attention basée sur la connaissance a priori prend en entrée la connaissance a priori et le résultat de l'opération de *pooling* appliquée à la sortie du CNN. Pour évaluer la valeur ajoutée de notre solution, nous avons considéré deux modèles similaires qui sont le *cnn-expert* et le *lstm-expert*. Toutefois, ces modèles n'ont pas la couche d'attention basée sur les connaissances a priori. Elles font uniquement une concaténation des données et du vecteur a priori. Elles nous servent d'éléments de comparaison avec les solutions proposées.

5.1.1.3 Résultats

Tous les modèles ont été entraînés sur 80% des données dont 20% ont été dédiés à la validation, les 20% restants ont été utilisés pour les tests. L'*accuracy* est une mesure standard de l'efficacité des modèles conçus. Cependant, pour des ensembles de données dont la distribution entre les classes à prédire est déséquilibrée comme la nôtre, cette mesure peut être illusoire et peu informative sur les erreurs commises par le classificateur. Au lieu de considérer seulement l'*accuracy*, nous

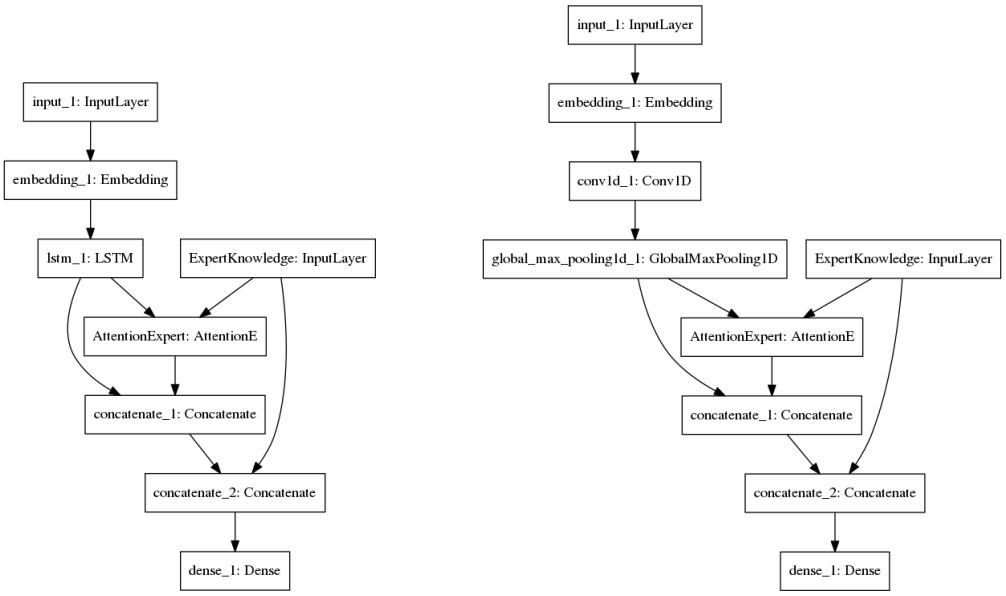


Figure 5.9 L'architecture hybride proposée utilisant un LSTM (à gauche) ou un CNN (à droite) pour la prédiction du niveau de raisonnement sociomoral.

avons considéré le rappel, la précision et la F-mesure qui prend en considération à la fois la précision et le rappel, pour toutes les classes et pour chacune des classes séparément.

Les résultats sont présentés dans le tableau 5.2 et dans les figures 5.10 et 5.11. Comme nous pouvons le constater, les modèles qui tiennent compte des connaissances a priori (des experts) donnent de bons résultats pour la prédiction des classes avec peu d'échantillons comparativement aux modèles qui n'ont pas tenu compte de ces connaissances (voir F-mesure pour les classes 2, 4, et 5). Ceci est principalement dû au fait que les experts n'ont pas besoin de données pour prendre des décisions parce que leurs décisions sont fondées sur leurs propres connaissances. Cela suggère que l'intégration des connaissances des experts aux modèles neuronaux améliore la classification même lorsque l'ensemble de données est déséquilibré. De plus, même s'il n'y a pas beaucoup de données pendant l'entraînement, les modèles sont tout de même capables de mieux généraliser sur des

données de test. Dans l'ensemble, les modèles utilisant le CNN ont mieux fonctionné par rapport aux modèles utilisant les LSTMs puisqu'il a été démontré que ce dernier est une solution qui a un meilleur pouvoir de généralisation quand il y a beaucoup de données disponibles. Nous avons donc intégré dans le jeu Les-Dilemmes, le modèle basé sur le CNN. La prédiction du niveau de raisonnement dans le jeu est faite en temps réel. Nous enregistrons tout d'abord la justification audio de l'apprenant avant de la transformer en texte à l'aide de l'API *Google Speech to text*[5]. Le texte est ensuite introduit dans le modèle qui fait la prédiction et met à jour le modèle de l'apprenant qui conserve toutes ces informations.

Tableau 5.2 Précision, Rappel, F-mesure et *accuracy* de tous les modèles.

Modèles	**Precision**	**Rappel**	**F-mesure**	***Accuracy***
Expert-Only	0.47	0.40	0.38	0.40
cnn-only	0.58	0.53	0.49	0.53
lstm-only	0.42	0.43	0.42	0.43
cnn-expert	0.62	0.62	0.62	0.62
lstm-expert	0.54	0.53	0.51	0.53
cnn-expert-att	0.67	0.65	0.63	0.65
lstm-expert-att	0.59	0.60	0.58	0.60
Modèle final	**0.72**	**0.75**	**0.73**	**0.75**

5.1.2 Évaluation subjective

Le but ici est de faire une évaluation basée sur les réponses fournies par les joueurs aux questionnaires, avant et après le jeu. Dans le questionnaire avant le jeu, nous évaluons le style d'apprentissage du joueur alors que dans le questionnaire posé après le déroulement du jeu, nous évaluons 3 dimensions à savoir l'apprentissage, l'immersion et la satisfaction. Nous avons donc évalué ces 3 dimensions tant sur le groupe ayant joué à la version non adaptative que sur le groupe ayant joué à la version adaptative. Les résultats obtenus dans cette section seront complétés

5. https ://cloud.google.com/speech-to-text/

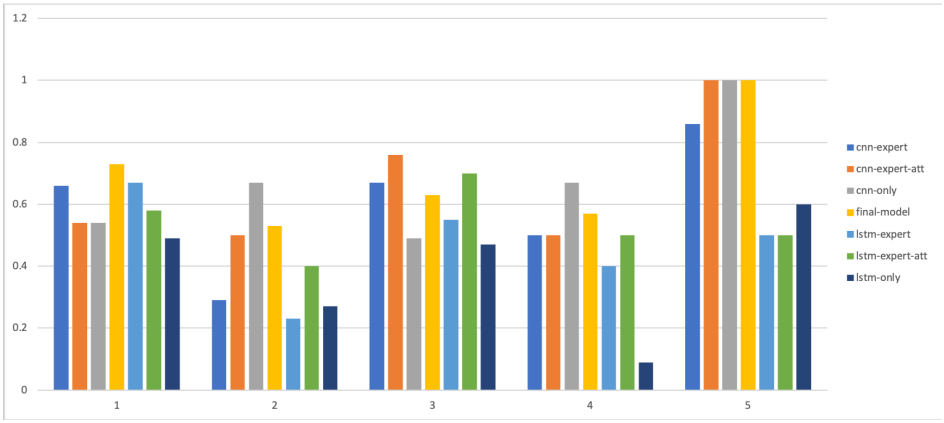


Figure 5.10 Précision de tous les modèles pour chaque niveau de raisonnement sociomoral.

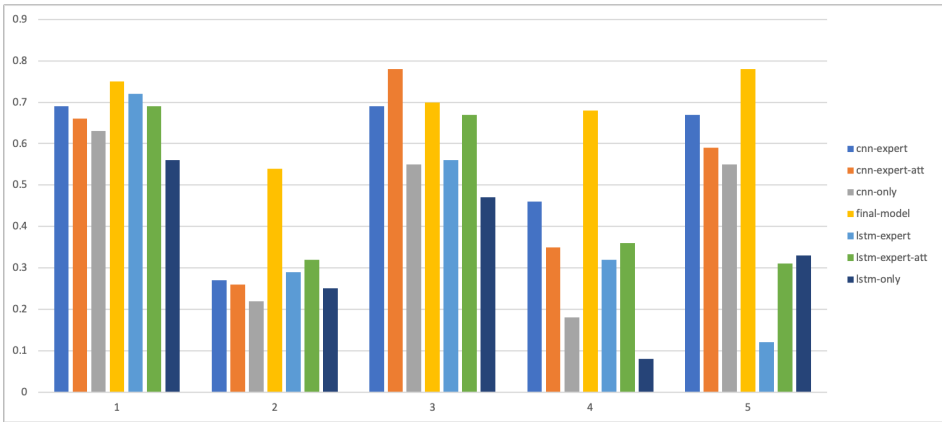


Figure 5.11 F-mesure de tous les modèles pour chaque niveau de raisonnement sociomoral.

par ceux obtenus de manière objective, qui seront présentés dans la section 5.1.3 suivante.

5.1.2.1 Évaluation de l'apprentissage

L'évaluation subjective de l'apprentissage vise à savoir ce que pense le joueur de son apprentissage dans le jeu. Des questions explicites telles que : *As-tu eu l'impression d'avoir appris des choses dans le jeu ?* lui ont été directement demandées. Les choix de réponses étaient présentés sous la forme d'une échelle de *Likert* allant

de 1 à 6 correspondant respectivement à ne rien avoir appris et totalement avoir appris. L'ensemble des questions posées pour l'évaluation de l'apprentissage est disponible en annexe B.2.

Les figures 5.12 et 5.13 présentent la différence de moyenne du taux d'apprentissage perçu selon les joueurs, entre les deux versions du jeu. Les moyennes des notes données par les joueurs ont été calculées. On peut voir dans la figure 5.12 qu'il y a une différence d'apprentissage perçue entre les deux versions (4.12/6 pour la version adaptative versus 3.54/6 pour la version non adaptative) même si elle n'est pas vraiment significative ($p = 0.064$ sachant que c'est significatif lorsque $p < 0.05$) comme on peut le voir dans la figure 5.13.

5.1.2.2 Évaluation de l'immersion

L'évaluation subjective de l'immersion vise à savoir ce que pense le joueur de l'immersion dans le jeu. Des questions explicites telles que : *J'étais parfois tellement impliqué que j'oubliais que j'étais dans un jeu.* lui ont été posé avec comme choix de réponses une échelle de *Likert* allant de 1 à 6 correspondant respectivement à parfait désaccord et parfait accord. Le questionnaire complet est disponible en annexe B.2.

Les figures 5.12 et 5.13 présentent la différence de moyenne du taux d'immersion selon le point de vue des joueurs, entre les deux versions du jeu. On peut voir dans la figure 5.12 qu'il y a une différence significative ($p = 0.045$) d'immersion entre les deux versions (4.04/6 versus 3.64/6) comme on peut le voir dans la figure 5.13. Ce qui veut dire que la version adaptative est significativement plus immersive que la version non adaptative. Notons, que dans la version adaptative, la musique de fond pouvait changer en fonction des émotions, ce qui est l'un des facteurs qui a probablement contribué à ce résultat.

5.1.2.3 Évaluation de la satisfaction

L'évaluation subjective de la satisfaction (efficacité du jeu) vise à évaluer ce que pense le joueur du jeu en général. Des questions explicites telles que : *As-tu aimé jouer à ce jeu ?* lui ont été posé avec comme choix de réponse une échelle de *Likert* allant de 1 à 6 correspondant respectivement à parfait accord et parfait désaccord. L'ensemble des questions posées pour l'évaluation de la satisfaction est également disponible en annexe B.2.

Les figures 5.12, 5.13 et 5.14 présentent la différence de moyenne du taux de satisfaction/efficacité selon les joueurs, entre les deux versions du jeu. Encore une fois, on peut voir dans la figure 5.13 qu'il y a une différence de satisfaction entre les deux versions (3.72/6 versus 3.15/6 pour la version non adaptative). Cependant, ce résultat n'est pas significatif puisque $p = 0.10$. Nous ne pouvons donc tirer aucune conclusion pour cette dimension. Peut-être qu'avec plus de participants on verrait mieux la différence. Plus d'investigations restent à faire à ce niveau.

Statistiques de groupe

	Adaptative	N	Moyenne	Ecart type	Moyenne erreur standard
Immersion	OUI	43	4,0407	,70274	,10717
	NON	23	3,6359	,87249	,18193
Satisfaction	OUI	43	3,7287	1,43972	,21955
	NON	23	3,1594	1,15741	,24134
Apprentissage	OUI	43	4,1296	1,15966	,17685
	NON	23	3,5404	1,30715	,27256

Figure 5.12 Statistiques des groupes. Les réponses ont été moyennées par catégorie de questions (immersion/apprentissage/satisfaction).

Test des échantillons indépendants

		Test de Levene sur l'égalité des variances		Test t pour égalité des moyennes						
									Intervalle de confiance de la différence à 95 %	
		F	Sig.	t	ddl	Sig. (bilatéral)	Différence moyenne	Différence erreur standard	Inférieur	Supérieur
Immersion	Hypothèse de variances égales	,272	,603	2,048	64	,045	,40483	,19771	,00985	,79980
	Hypothèse de variances inégales			1,917	37,549	,063	,40483	,21114	-,02278	,83244
Satisfaction	Hypothèse de variances égales	2,088	,153	1,633	64	,107	,56926	,34858	-,12710	1,26562
	Hypothèse de variances inégales			1,745	54,081	,087	,56926	,32626	-,08483	1,22336
Apprentissage	Hypothèse de variances égales	2,066	,156	1,881	64	,064	,58920	,31319	-,03648	1,21487
	Hypothèse de variances inégales			1,813	40,649	,077	,58920	,32491	-,06714	1,24553

Figure 5.13 Comparaison entre l'évaluation subjective de la version non adaptative et l'évaluation subjective de la version adaptative du jeu, en utilisant une comparaison des moyennes dont le test T avec échantillons indépendants.

5.1.3 Évaluation objective

L'objectif ici est de faire une évaluation basée sur les réactions et les réponses des joueurs pendant le déroulement du jeu. L'évaluation est faite en comparant le niveau de raisonnement atteint par les joueurs dans la version adaptative et la version non adaptative du jeu. Nous évaluerons le taux l'apprentissage et l'impact des règles d'adaptation sur les joueurs, en particulier celles liées aux réactions émotives. Puisque les règles en général ont été conçues pour l'amélioration de l'apprentissage, elles seront donc intrinsèquement évaluées lors de l'évaluation de l'apprentissage.

5.1.3.1 Évaluation de l'apprentissage

Afin de mesurer le potentiel du jeu dans le support des usagers à développer un niveau plus élevé de maturité sociale, nous allons comparer les résultats (moyenne des niveaux de raisonnement) qu'ont obtenus les joueurs-apprenants pendant le pré-test, le post-test et le jeu. Nous avons éliminé les participants dont les données étaient partielles et ou biaisées : les justificatifs n'ont pas pu être transcrits correctement ce qui a impliqué une cotation qui ne reflétait pas le niveau actuel dans le jeu. Un pré-test et un post-test ont été utilisés pour comparer les niveaux de

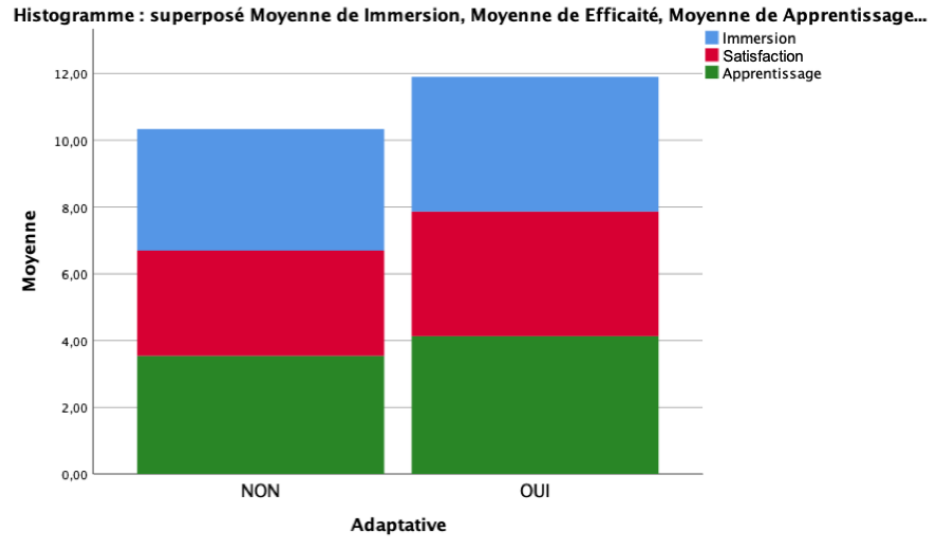


Figure 5.14 Visualisation de la différence (comparaison des moyennes) entre la version non adaptative (NON) et adaptative du jeu (OUI).

maturité avant et après avoir joué. Les résultats (figures 5.15 et 5.16) montrent que le niveau de raisonnement moyen des joueurs pendant le pré-test (1.70) est plus bas que leurs niveaux de raisonnement moyen pendant le jeu (2.30) et le post-test (2.53) de façon très significative ($p = .001$ et $p = 0.02$). Ainsi, le jeu a permis un gain notable de niveau de raisonnement qui on peut le dire, a persisté même après le jeu : puisque le niveau de raisonnement pendant le post-test est très peu ($p = 0.44$) supérieur à celui du jeu.

Nous avons également fait une comparaison entre la version non adaptative et la version adaptative du jeu. Pour cette évaluation (voir figures 5.17 et 5.18), nous avons ajouté une variable (appelée **Pre_Pro**) mesurant la différence entre le pré-test et le post-test ($post-test - pr-test$) pour chaque participant. Comme nous pouvons le voir dans ces figures, la différence entre le post-test et le pré-test est plus élevée ($p = 0.058$) pour ceux qui ont joué à la version adaptative (.7889 versus 0.333) comparativement à ceux qui ont joué la version non adaptative. Notons que les niveaux de raisonnement avant de commencer à jouer (pré-test) sont

Statistiques des échantillons appariés

		Moyenne	N	Ecart type	Moyenne erreur standard
Paire 1	Prétest	1.70000000	15	.574594405	.148359637
	Postest	2.48888889	15	.679830355	.175531443
Paire 2	Jeu	2.30338542	16	.724390081	.181097520
	Postest	2.53645833	16	.683786266	.170946566
Paire 3	Prétest	2.08000000	25	.929356607	.185871321
	Jeu	2.57064286	25	.774129651	.154825930

Figure 5.15 Évaluation de l'apprentissage dans le jeu LesDilemmes : Statistiques de groupe après un test T pour échantillons appariés.

Test des échantillons appariés

		Différences appariées					t	ddl	Sig. (bilatéral)
		Moyenne	Ecart type	Moyenne erreur standard	Intervalle de confiance de la différence à 95 % Inférieur	Supérieur			
Paire 1	Prétest - Postest	-.78888889	.705608672	.182187376	-1.1796419	-.39813583	-4.330	14	.001
Paire 2	Jeu - Postest	-.23307292	1.18842413	.297106032	-.86633943	.400193600	-.784	15	.445
Paire 3	Prétest - Jeu	-.49064286	1.05640997	.211281994	-.92670746	-.05457825	-2.322	24	.029

Figure 5.16 Évaluation de l'apprentissage dans le jeu LesDilemmes : Résultats du test T pour échantillons appariés.

significativement plus bas ($p = 0.02$) pour ceux qui ont joué à la version adaptative (2.17 versus 1.7) comparativement aux autres. Ceci s'explique par le fait que les participants qui ont joué à la version non adaptative étaient en moyenne plus agés que ceux qui ont joué à la version adaptative. Ainsi, la version adaptative du jeu a été plus efficace en matière de support à l'apprentissage du raisonnement sociomoral comparé à sa version non adaptative. Cependant, nous sommes persuadé que ces résultats pourront encore s'améliorer lorsque le modèle qui prédira le niveau de raisonnement sera plus fidèle et que la transcription de l'audio vers le texte sera plus exacte (ce qui n'est actuellement pas le cas).

5.1.3.2 Évaluation des règles d'adaptation

Comme mentionné ci-dessus, nous allons uniquement évaluer les règles qui visent à garder les joueurs dans un état émotif positif (ex : ***SI*** *valence <0* ***ALORS***

Statistiques de groupe

	Adaptive	N	Moyenne	Ecart type	Moyenne erreur standard
Pre_Pro	A	15	.7889	.70561	.18219
	NA	29	.3333	.74934	.13915
Prétest	A	15	1.70000000	.574594405	.148359637
	NA	29	2.17816092	.675434949	.125425121
Postest	A	16	2.53645833	.683786266	.170946566
	NA	29	2.51149425	.648458236	.120415671

Figure 5.17 Comparaison des 2 versions du jeu LesDilemmes : Statistiques de groupe après un test T pour échantillons appariés.

Test des échantillons indépendants

		Test de Levene sur l'égalité des variances		Test t pour égalité des moyennes						
									Intervalle de confiance de la différence à 95 %	
		F	Sig.	t	ddl	Sig. (bilatéral)	Différence moyenne	Différence erreur standard	Inférieur	Supérieur
Pre_Pro	Hypothèse de variances égales	.095	.759	1.949	42	.058	.45556	.23378	-.01622	.92733
	Hypothèse de variances inégales			1.987	29.994	.056	.45556	.22925	-.01263	.92375
Prétest	Hypothèse de variances égales	.400	.531	-2.336	42	.024	-.47816092	.204683987	-.89122993	-.06509191
	Hypothèse de variances inégales			-2.461	32.789	.019	-.47816092	.194273115	-.87350917	-.08281267
Postest	Hypothèse de variances égales	.101	.752	.121	43	.904	.024964080	.205847878	-.39016773	.440095890
	Hypothèse de variances inégales			.119	29.666	.906	.024964080	.209099647	-.40227614	.452204304

Figure 5.18 Comparaison des 2 versions du jeu LesDilemmes : Résultats du test T pour échantillons appariés.

musique joviale sinon musique douce), puisque ce dernier favorise l'apprentissage (Um *et al.*, 2012; Tyng *et al.*, 2017). Les émotions permettent également de maximiser l'engagement de l'apprenant et améliorer son apprentissage et sa rétention à long terme (Shen *et al.*, 2009). Pour ce faire, pour tous les participants des 2 expérimentations, nous avons fait une moyenne de leurs réactions émotives (calculées à partir du Facereader) par rapport à chaque dilemme répondu. Nous avons considéré les 7 émotions de base de Ekman (1970) à savoir : le neutre (*Neutral*), la joie (*Happy*), la tristesse (*Sad*), la colère (*Angry*), la surprise (*Suprised*), la peur (*Scared*) et le dégoût (*Disgusted*), ainsi que la *valence* et l'*arousal*. Ces deux dernières

sont calculées selon des formules bien définies [6], et toutes les valeurs varient entre 0 et 1 sauf la valence qui varie de -1 à 1. Il est à noter que, les émotions *neutre* et *colère* ont tendance à être plus présentes dans les activités impliquant la lecture. De plus, ces émotions ont tendance à se manifester avec une plus grande intensité que les autres lorsque capturées avec le Facereader (Terzis *et al.*, 2010; Alitalo, 2016). Dans le jeu LesDilemmes, l'activité la plus présente est la lecture puisque les joueurs doivent lire et écouter les avis des autres et les consignes du jeu. Ceci est encore plus vrai dans la version adaptative du jeu puisque nous avons rajouté des messages d'apprentissage, de félicitations, etc. Également, le jeu en tant que tel n'a pas une dynamique d'un «vrai jeu» dans le sens où le personnage principal n'a aucun pouvoir sur l'environnement à part prendre des décisions, donner son avis et évaluer les autres à travers des clics sur des boutons. Ainsi, on observera en général des émotions plus négatives que positives, ce qui est tout à fait justifié.

Nous avons fait une première comparaison des moyennes entre la valence et l'arousal (voir figure 5.19) sur les 2 versions du jeu. On peut voir que la valence moyenne est significativement plus grande dans la version adaptative (A) que dans la version non adaptative (NA). La version adaptative a donc suscité ($p < 0.001$ voire tableau E.2 en annexe) plus d'émotions positives que la version non adaptative malgré la présence d'un contenu plus «textuel». Ce qui implique que les règles visant à mettre l'apprenant dans un état plus positif ont eu de l'effet. Bien sûr, on remarque que la valence dans les 2 cas est négative ce qui s'explique par le fait que le jeu implique beaucoup de lecture qui suscite des émotions négatives. Nous avons fait une deuxième comparaison des moyennes (voir figure 5.20) impliquant quelques des sept émotions de base dont le $p < 0.001$. On peut constater de cette figure que la version non adaptative a suscité significativement plus de colère et de dégoût que la version adaptative ($p < 0.001$). Par contre la version adaptative

6. https ://pdfs.semanticscholar.org/2155/739f578e33449546f45a0b4cf64dbd614025.pdf

a suscité plus du neutre et de surprise chez les joueurs, comparée à la version non adaptative. Plusieurs recherches ont montré que la surprise est une émotion jouant un rôle majeur dans l'apprentissage. Oudeyer et al (Oudeyer *et al.*, 2016) ainsi que Meadhbh et al (Foster et Keane, 2019) ont montré que les éléments provoquant la surprise sont conservés en mémoire plus facilement et sont rappelés plus précisément que les éléments provoquant moins de surprise (éléments prévisibles). Adler (Adler, 2008) a conclu que la surprise est une émotion d'une grande valeur dans l'apprentissage ; lorsque les apprenants rencontrent une information surprenante, leur attention est attirée (c.-à-d. qu'ils remarquent la surprise), ce qui provoque un traitement plus intensif du matériel à apprendre (c.-à-d. la résolution) ; il faut donc corriger et mieux comprendre ce matériel). Munnich et al (Munnich *et al.*, 2007), ont montré que la surprise favorise la rétention, peut-être parce que la surprise peut rendre un événement plus intéressant et agréable (Loewenstein et Heath, 2009). Les points de vue connexes sont repris dans le domaine de l'IA, où la surprise a été proposée comme mécanisme cognitif pour identifier les événements qui sont des opportunités d'apprentissage dans les architectures d'agents pour robots (Bae et Young, 2009; Macedo *et al.*, 2009; Macedo et Cardoso, 2012). Ainsi, le fait que la version adaptative ait suscité plus de surprise, implique qu'elle a eu plus d'effet et ou aura plus d'effet à long terme comparée à la version non adaptative.

Les tableaux E.1 et E.2 situés en annexe présentent plus en détail les résultats obtenus pour l'analyse sur les émotions capturées dans les 2 versions du jeu.

5.1.3.3 Évaluation du modèle de prédiction du raisonnement sociomoral en temps réel dans le jeu

L'objectif de cette étape est d'évaluer la performance en temps réel du modèle de prédiction du niveau de raisonnement dans le jeu. Nous avons conduit deux analyses dont la première est une comparaison entre les prédictions faites par le

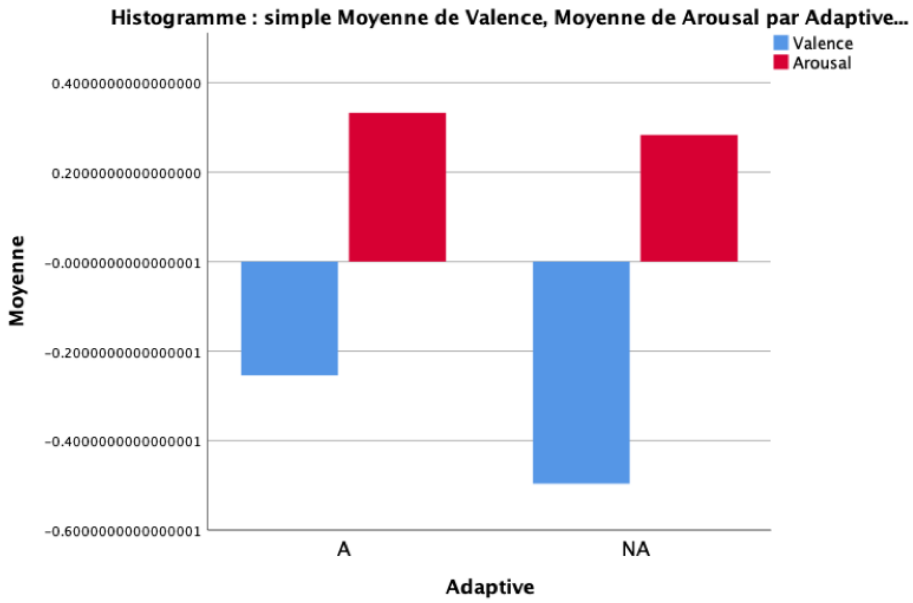


Figure 5.19 Visualisation de la comparaison des moyennes de la valence et de l'arousal ($p<0.001$) entre la version non adaptative (NA) et adaptative (A) du jeu.

modèle et les annotations des experts. La deuxième est la comparaison entre les prédictions faites par le modèle automatique à partir de transcription audio vers texte (des verbatims) effectuées par les humains et non pas par un système automatique comme c'est fait présentement dans le jeu, et les annotations des experts. Cette deuxième analyse vise à évaluer l'impact de la transcription automatique sur la performance du modèle de prédiction du niveau de raisonnement. Pour ces analyses, nous avons considéré que, si $|R_p - R_r| \leq 1$, alors la prédiction est correcte, avec R_p et R_r représentant respectivement le niveau de raisonnement prédit et le niveau de raisonnement réel. Cette marge d'erreur se base sur le fait que les experts qui font la cotation, ont tendance à attribuer pour un même verbatim, des niveaux de raisonnement jusqu'à plus ou moins 1 de différence. De plus, l'expérience a montré que deux experts différents ne feront pas toujours la même cotation pour un même verbatim. Ainsi, la comparaison entre les prédictions dans

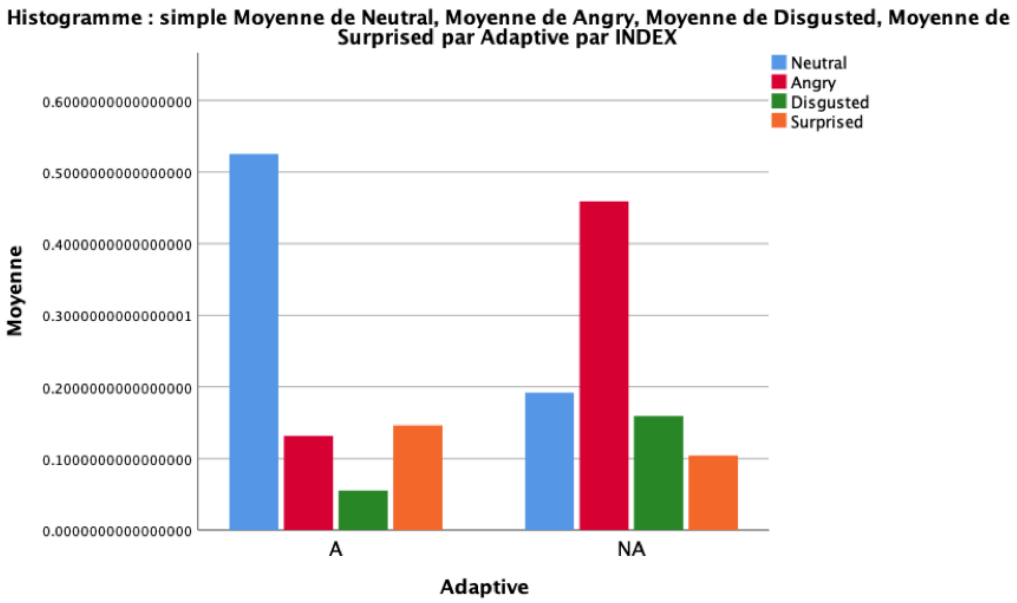


Figure 5.20 Visualisation de la différence (comparaison des moyennes) des émotions (p <0.001) entre la version non adaptative (NA) et adaptative (A) du jeu.

le jeu et les cotations réelles des verbatims émis pendant le jeu, a donné une précision de 69%. En d'autres termes, les deux cotations sont semblables à 69%, ce qui est largement au-dessus du hasard. Pour ce qui est de la deuxième analyse où R_p représente la valeur du niveau de raisonnement prédit par le modèle à partir des verbatims transcrits par les humains, la précision obtenue a été de 76.6% qui est meilleure que celle obtenu dans la première analyse. Nous voyons que les prédictions sont meilleures sur les verbatims transcrits par un humain que ceux transcrits automatiquement pendant le jeu. La transcription automatique est donc un point important à surveiller lors des futures versions du jeu, puisqu'elle est la principale cause de la moins bonne performance du modèle pendant le jeu.

En conclusion de cette partie, l'évaluation du jeu suggère qu'il a été apprécié par les joueurs en matière d'immersion, de jouabilité et d'impression d'avoir appris quelque chose. Les résultats montrent également que le jeu encourage le dévelop-

pement de niveaux plus élevés de maturité de raisonnement. Également, le jeu dans sa version adaptative permet de garder le joueur dans un état émotionnel propice à la persistance de la connaissance apprise mais également au plaisir d'apprendre. Bien qu'il reste encore à améliorer le jeu en tant que tel (la dynamique du jeu), notre algorithme de prédiction du niveau de raisonnement sociomoral et la transcription audio vers texte, on peut néanmoins conclure que la version adaptative a eu un effet significativement plus positif que la version non adaptative sur toutes les dimensions évaluées. Nous sommes persuadés que cet impact positif serait encore plus élevé si la transcription automatique de l'audio vers le texte était plus précise. Une autre possibilité serait de passer à l'anglais, ce qui est déjà prévu pour les prochaines versions du jeu, puisque la transcription automatique de l'audio vers le texte lorsqu'il s'agit de l'anglais est plus développé que le francais. Cette option nécessiterait de ré-implémenter le jeu au complet en anglais et d'avoir à disposition des verbatim en anglais et annotés par les experts afin pouvoir d'entraîner de nouveau le modèle de cotation automatique.

5.2 Muse-logique : un système intelligent pour l'apprentissage du raisonnement logique

Muse-logique (voir figures G.3 à G.2 en annexe) est un système tutoriel intelligent capable de modéliser et simuler le raisonnement humain tout en expliquant son cheminement, et de faire apprendre aux humains à être de meilleurs raisonneurs (Nkambou *et al.*, 2015; Tato, 2016). Muse-logique se veut être un outil extensible à d'autres types de raisonnements et d'autres logiques. Le modèle tuteur (Muse-tuteur) joue plusieurs rôles, entre autres la conduite de l'interaction avec les apprenants, la détection les erreurs de l'apprenant et y remédier en utilisant les règles prédéfinies par les experts du domaine, le diagnostic cognitif des erreurs détectées, etc. Le modèle expert (Muse-expert) a pour but de produire

des raisonnements valides dans toutes les situations qui lui sont soumises et d'expliquer son processus de raisonnement. Il est capable de détecter des erreurs de raisonnement et vérifier les actions légales dans le contexte suivant les règles qui le régissent, suivre des schémas de raisonnement dans le but de les valider et d'en proposer des correctifs. Il collabore avec le tuteur pour l'aider à évaluer et valider les productions de l'apprenant, et à déterminer ses compétences logiques. Le modèle d'inférence (valide et invalide) est codé en tant que règles de production, et la mémoire sémantique de la logique ciblée est codée dans une ontologie formelle OWL et reliée aux règles d'inférence. Finalement, le modèle apprenant qui modélise l'état de connaissances en raisonnement logique de l'apprenant. Les connaissances de l'apprenant sont modélisées comme un recouvrement (overlay) sur les connaissances de l'expert, mais à l'aide d'un réseau bayésien. Ledit réseau bayésien (voir figure G.1 en annexe) a été conçu entièrement par les experts : de l'architecture à la définition des probabilités a priori. Le réseau est mis à jour à chaque réponse de l'apprenant puisque les nœuds observables ou évidences sont directement connectés aux exercices alors que les nœuds latents représentent les connaissances qui ne sont pas directement observables. Les nœuds latents représentent des compétences acquises ou non par l'apprenant, qui seront observées par le système. Ces nœuds ont une probabilité continue comprise entre 0 et 1. Les nœuds des évidences représentent les réponses de l'apprenant aux exercices proposés par le système. Les nœuds des évidences sont représentés par une variable aléatoire Q avec une distribution de Bernoulli. La valeur $Q = 1$ signifie que l'étudiant a répondu correctement à l'exercice et $Q = 0$ signifie que sa réponse est incorrecte. Ainsi, lorsque l'apprenant répond à un exercice, les probabilités de maîtrise des différentes connaissances sont mises à jour. Le tuteur pour le choix du prochain exercice à présenter à l'usager, interroge le réseau pour extraire les connaissances encore mal maîtrisées. C'est donc dans cette liste de connaissances que le tuteur sélectionne le ou les exercices qui doivent être proposés à l'usager.

5.2.1 Évaluation du modèle de l'usager

Les résultats présentés dans cette section ont fait l'objet de publications acceptées (voir section 5.4.1). La première version de Muse-logique se concentre sur la logique propositionnelle. L'objectif du modèle apprenant de Muse-logique est de représenter, mettre à jour et prédire l'état des connaissances de l'apprenant en fonction de son interaction avec le système. Il comporte de multiples aspects dont la partie cognitive qui représente essentiellement l'état des connaissances de l'apprenant : maîtrise des capacités de raisonnement dans chacune des six situations de raisonnement identifiées grâce aux experts. L'état cognitif est généré à partir du comportement de l'apprenant lors de ses interactions avec le système, c'est-à-dire qu'il est déduit par le système à partir des informations disponibles. Il est soutenu par un réseau bayésien (Tato *et al.*, 2016; Tato *et al.*, 2017a) basé sur la connaissance du domaine, où les relations d'influence entre les nœuds ainsi que les probabilités préalables sont fournies par les experts. Certains nœuds sont directement liés aux activités de raisonnement telles que les exercices. Les connaissances impliquées dans le système sont celles mises en avant par la théorie des modèles mentaux pour raisonner en conformité avec les règles logiques. Il y a 16 connaissances directement observables c'est-à-dire liées aux exercices et 12 connaissances latentes. Il y a un total de 48 exercices dont 3 par item de connaissances observables.

Actuellement, le traçage des connaissances dans Muse-logique se fait uniquement via ce réseau bayésien. Ayant déjà ce modèle qui fonctionne bien (Accuracy de 65%), nous avons choisi de l'utiliser comme connaissance experte (a priori) pour la mise en place d'un nouveau modèle de traçage de connaissances hybride plus performant, qui le fusionne avec le DKT. Pour ce faire, nous utilisons notre nouvelle version du DKT que l'on combine au réseau bayésien en utilisant notre solution hybride (la figure 5.21 présente l'architecture complète). Nous avons conduit

des tests qui visent à évaluer la capacité de traçage de connaissance du nouveau modèle versus le réseau bayésien. Il est à noter que ces tests ont été faits uniquement sur les données récoltées mais pas en temps réel dans le système. Nous avons utilisé AAdam (la version accélérée de Adam) comme algorithme d'optimisation de l'apprentissage.

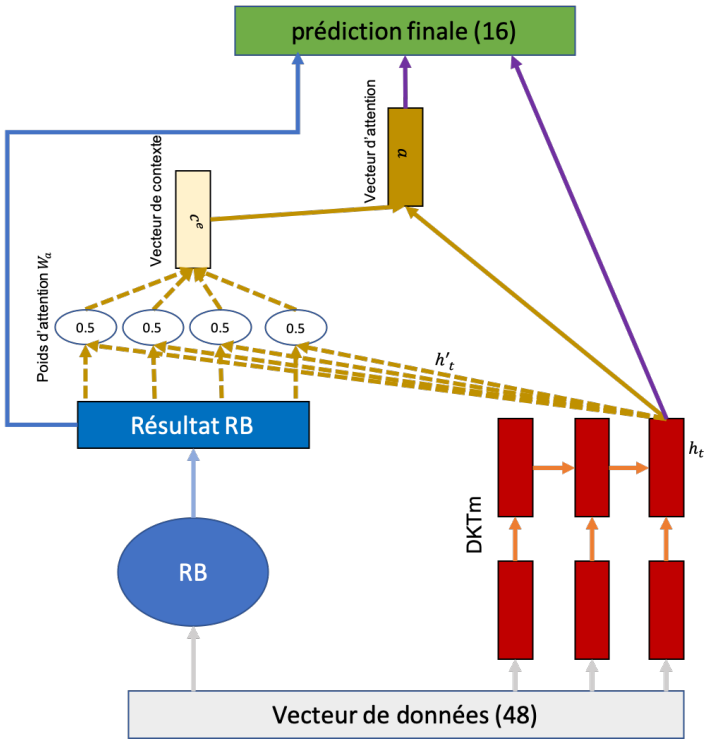


Figure 5.21 Modèle DKT hybride — À chaque instant t, le modèle déduit un vecteur de poids d'alignement basé sur les connaissances actuelles prédites par le DKTm (y_t) et toutes les entrées du vecteur de connaissances expert. Le vecteur de contexte côté expert c^e est ensuite calculé comme la moyenne pondérée, sur chaque entrée du vecteur de connaissances expert.

5.2.1.1 Le jeu de données

Un total de 294 participants ont participé à cette étude. Ils ont tous complété les 48 exercices de raisonnement logique. Dans notre ensemble de données, chaque ligne de données représente chaque participant (un total de 294 entrées et une

Tableau 5.3 Distribution des réponses par rapport aux connaissances — Connaissances difficiles à maîtriser (moyenne < 0.4) et connaissances faciles à maîtriser (moyenne > 0.9) sont en gras.

Connaissances	N	Moyenne	Écart type
MppFd	294	**0,9456**	0,16078
MppMd	294	0,898	0,23726
MppCcf	294	**0,907**	0,2394
MppA	294	**0,9615**	0,16066
MttFd	294	0,8435	0,26646
MttMd	294	0,7925	0,29985
MttCcf	294	0,7494	0,33326
MttA	294	0,8401	0,28974
AcMa	294	0,424	0,38072
AcFa	294	**0,3039**	0,3652
AcCcf	294	**0,3345**	0,40801
AcA	294	**0,2823**	0,41038
DaMa	294	0,407	0,37389
DaFa	294	**0,3027**	0,35081
DaCcf	294	**0,381**	0,40662
DaA	294	**0,305**	0,42077

longueur de séquence de 48). Cette quantité de données est très limitée pour entraîner un modèle d'apprentissage profond. Cependant, combinés aux connaissances des experts, nous verrons une différence substantielle dans les résultats. Les exercices ont été encodés en utilisant les 16 connaissances directement observables, ce qui signifie que les questions relatives à la même connaissance sont encodées avec le même Id ($1 \sim 16$). Les connaissances avec peu de données d'entraînement (connaissances rares) sont déterminées par une comparaison de la moyenne des bonnes réponses obtenues pour chaque connaissance. Dans le tableau 5.3, nous avons fait la moyenne de toutes les réponses pour chaque connaissance. Les connaissances difficiles sont celles dont les valeurs moyennes sont les plus basses et les connaissances faciles sont celles dont les moyennes sont les plus élevées. La deuxième et la dernière partie de notre nouvelle fonction de perte portent donc sur ces connaissances. Puisque le RNN n'accepte qu'une longueur fixe de vecteurs

en entrée, nous avons utilisé un codage *one-hot* pour convertir la performance des apprenants en une longueur fixe de vecteurs dont tous les éléments sont à 0 sauf un seul qui est à 1. Le 1 dans ce vecteur indique deux choses : quelle connaissance a été évaluée et si la question portant sur cette connaissance a été bien répondue ou pas.

5.2.1.2 Expériences et résultats

Nous allons maintenant évaluer la solution DKT et le modèle hybride proposé pour le traçage de connaissances dans muse-logique. Nous avons utilisé 3 modèles : le DKT, le DKT où nous avons appliqué un masque à la fonction de perte (DKTm) et le DKTm hybride avec connaissances a priori (DKTm+BN). Nous avons utilisé 20 % des données pour les tests et 15 % pour la validation. Comme mentionné ci-dessus, le BN à lui seul a une performance de 65 % de précision globale sur ces données. Le résultat est évalué à l'aide du F-mesure sur chaque connaissance (traitée comme 2 classes - réponses correctes et incorrectes) en jeu et sur la précision globale du modèle. Les modèles ont été évalués sur 20 expériences différentes et la moyenne des résultats finaux a été calculée. Dans toutes nos expériences, nous avons fixé λ_1, $\lambda_2 = 0.10$. Notre implémentation du modèle DKT avec Keras a été inspirée de l'implémentation [7] faite par Khajah et al. (Khajah *et al.*, 2016).

Les résultats sont présentés dans les tableaux 5.4 et 5.5 et dans la figure 5.22. La figure 5.22 montre une visualisation du traçage en temps réel des connaissances en raisonnement logique d'un apprenant. L'apprenant répond d'abord correctement à une question relative à la connaissance **DA_FFA** (DA_FFA - 1) et à une question relative à la compétence **AC_FMA** (AC_FMA - 1) et ensuite échoue à la question portant sur la connaissance **MTT_FDD** (MTT_FDD - 0). Dans les 45 questions suivantes, l'apprenant résout une série de problèmes liés aux 16

7. https ://github.com/mmkhajah/dkt

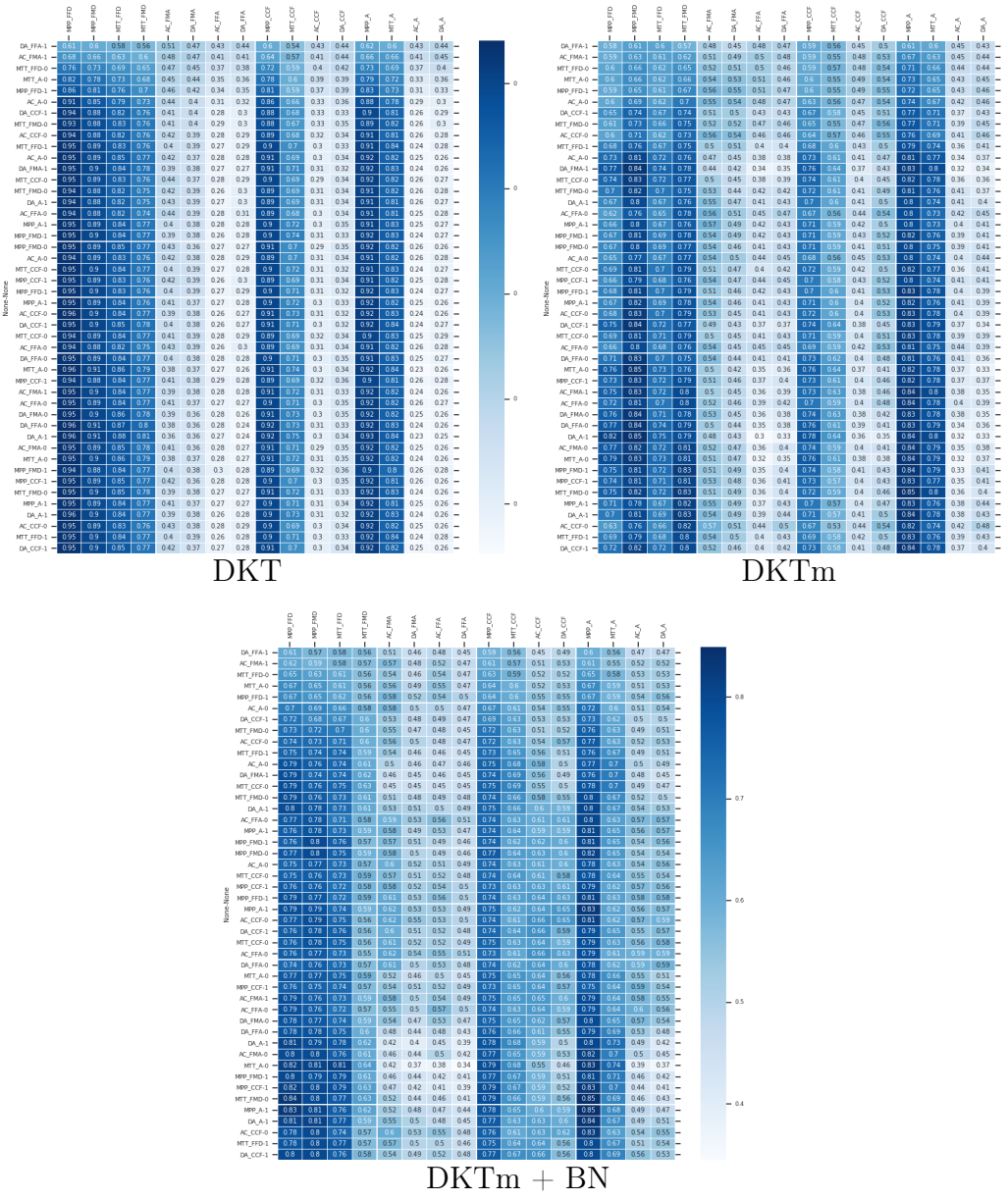


Figure 5.22 Carte de chaleur illustrant la prédiction avec les 3 modèles sur le même apprenant.—L'axe verticale représente la compétence mis en jeu dans la question posée à l'instant t suivis de la réponse réelle de l'apprenant (il y a 48 lignes représentant les 48 questions). Les couleurs (et les probabilités à l'intérieur de chaque point) indiquent la probabilité prédit par les modèles que l'apprenant répondra correctement à une question lié à une compétence (en horizontale) à l'instant suivant. Plus la couleur est foncée, plus la probabilité est élevée.

Tableau 5.4 Le DKT, le DKTm et le DKKm+BN sur les connaissances difficiles à maîtriser — Nous avons répété les expériences 20 fois. La dernière colonne est la précision globale du modèle. Pour chaque connaissance, la première colonne correspond à la valeur de la F-mesure pour la prédiction des réponses incorrectes et la deuxième colonne pour la prédiction des réponses correctes. La meilleure valeur pour chaque connaissance est en gras.

F-mesure	AC_FMA		DA_FMA		AC_FFA		DA_FFA		-
DKT	0.74	0.09	0.72	0.0	0.79	0.0	0.79	0.0	
DKTm	0.70	0.36	**0.74**	**0.55**	0.82	0.37	**0.80**	**0.47**	
DKTm+BN	**0.70**	**0.64**	0.78	0.43	**0.86**	**0.74**	**0.80**	**0.47**	
F-mesure	AC_CCF		DA_CCF		AC_A		DA_A		Acc
DKT	0.79	0.0	0.73	0.0	0.80	0.0	0.79	0.0	**0.74**
DKTm	0.83	0.54	0.76	0.51	0.85	0.48	0.86	0.46	**0.80**
DKTm+BN	**0.87**	**0.80**	**0.82**	**0.72**	**0.83**	**0.58**	**0.88**	**0.80**	**0.82**

Tableau 5.5 Le DKT, Le DKTm et le DKKm+BN sur les éléments de connaissances faciles à maîtriser. Les expériences ont été reprétées 20 fois.

F-mesure	MPP_FFD		MPP_FMD		MTT_FFD		MTT_FMD		MPP_CCF		MTT_CCF		MPP_A		MTT_A	
DKT	0.0	0.97	0.0	0.94	0.0	0.91	0.0	0.90	0.0	0.96	0.0	0.87	0.0	0.96	0.0	0.90
DKTm	0.0	0.97	0.02	0.97	0.0	0.94	0.22	0.89	0.0	0.95	0.09	0.84	0.0	0.99	0.0	0.90
DKTm+BN	0.0	0.95	0.0	0.97	0.0	0.89	0.19	0.88	0.0	0.97	0.12	0.85	0.0	0.99	0.1	0.89

connaissances impliquées. Chaque fois que l'apprenant répond à un exercice, les modèles mettent à jour ses connaissances de façon à pouvoir prédire s'il répondra correctement ou non à un exercice portant sur chaque connaissance lors de sa prochaine interaction. Dans la visualisation, nous ne montrons que les prédictions dans le temps pour les 48 exercices. Dans cette figure on constate facilement que le DKT est incapable de faire des prédictions précises sur les compétences avec peu de données, surtout celles qui sont difficiles à maîtriser (voir la première image). Ici, l'apprenant a donné 2 bonnes réponses sur 3 sur la compétence AC_FMA (5e colonne) et nous voyons que le DKTm et le DKTm+BN sont capables de prédire cette information, comparativement au DKT.

Notre version du DKT (DKTm) surpasse le DKT initial pour le traçage de toutes les connaissances en jeu. De plus, le DKTm amélioré avec la connaissance experte en utilisant notre solution hybride surpasse tous les autres modèles en matière de capacité de prédiction avec peu de données. Pour les connaissances faciles à maîtriser (par exemple le **MPP**), tous les modèles prédisent toujours que les apprenants donneront des réponses correctes (le F-mesure des réponses incorrectes est 0 pour le DKT et presque 0 pour les deux autres modèles) même après l'application du masque sur la fonction de perte. Ceci est dû principalement au fait que sur toutes les données, nous n'avons par exemple que 6 réponses incorrectes pour la connaissance **MPP_FFD**. Nous avons testé les modèles avec des valeurs élevées pour λ_2 et nous avons obtenu des valeurs du F-mesure égales à environ 0.6 pour les bonnes réponses et à environ 0.7 pour les mauvaises réponses. Ce résultat peut être satisfaisant dans d'autres contextes, mais dans le contexte du raisonnement logique où il est établi que le **MPP** est une connaissance qui est toujours bien maîtrisée, obtenir 0.6 comme valeur du F-mesure pour la prédiction de réponses correctes n'est pas acceptable. C'est pourquoi nous avons gardé λ_2 =0.10. Cependant, la solution reste valable pour les données où le rapport nombre de bonnes réponses

/ nombre de questions répondues ou nombre de mauvaises réponses / nombre de questions répondues sur une compétence n'est pas trop bas et est inférieur à 0.5. Pour les connaissances difficiles à maîtriser, il existe une grande différence entre le DKT et les autres modèles. Le DKT n'est pas en mesure de prédire les bonnes réponses sur les connaissances difficiles à maîtriser. Ce comportement ne peut pas être toléré car le traçage des connaissances échouera pour les apprenants qui maîtrisent cette connaissance. Il est donc important de s'assurer que le modèle final soit précis pour toutes les connaissances. Sur les connaissances rares, on constate que le DKTm et le DKTm + BN se comportent mieux que le DKT, ce qui signifie que les changements que nous avons apportés au DKT original n'affectent pas sa capacité prédictive.

Ces résultats montrent que le DKT n'est pas en mesure de faire un traçage avec précision des connaissances difficiles et faciles, comparativement au DKTm et au DKTm+BN. Au cours de nos expériences, nous avons remarqué que, pour les connaissances très faciles à maîtriser, les trois modèles n'ont pas été en mesure de tracer les réponses incorrectes, en raison du fait que le rapport réponses incorrectes/total des réponses aux questions était très faible. Même avec une pondération de la fonction de perte, nous n'avons pas été en mesure d'améliorer significativement le modèle DKT sur les connaissances impliquées. Cependant, avec le modèle hybride, les résultats ont été notables. Nous avons fixé les valeurs des paramètres de régularisation λ_1 et λ_2, mais pour les travaux futurs nous prévoyons de faire une recherche par quadrillage (*grid search*) pour trouver les meilleures valeurs. Nous sommes également conscients que le manque de données pourrait biaiser les résultats obtenus puisque les architectures d'apprentissage profond fonctionnent mieux sur des ensembles de données plus importants. Cependant, le problème des connaissances rares peut toujours survenir même avec un grand ensemble de données. Ainsi, nous pensons que les solutions que nous avons

proposées peuvent parfaitement fonctionner sur des ensembles de données plus importants.

5.3 Autres projets applicatifs

Nos solutions ont été appliquées à d'autres contextes visant la modélisation de l'usager mais pas pour l'amélioration de l'apprentissage.

5.3.1 Prédiction des émotions

Les résultats présentées dans cette section ont fait l'objet d'une publication (Tato *et al.*, 2018b). La prédiction des émotions est importante entre autres pour la compréhension du comportement humain et pour la modélisation des usagers dans les environnements d'apprentissage. Nous présentons ici, une architecture multimodale profonde pour la prédiction des émotions. Cette architecture tire parti de l'apprentissage profond, des données multimodales des usagers et de la hiérarchie de la mémoire humaine (Atkinson et Shiffrin, 1968). L'architecture consiste principalement en la combinaison de LSTMs. Dans les études sur le fonctionnement du cerveau, la mémoire est souvent divisée en deux types principaux : la mémoire explicite et la mémoire implicite, cette dernière étant implémentée dans les architectures LSTMs. Le modèle résultant a été testé sur un ensemble de données multimodales publiques.

5.3.1.1 Modèle multimodal LSTM (DM-LSTM)

Le DM-LSTM (deep multimodal LSTM) que nous proposons inclut les deux types de mémoires à long terme selon la théorie de la mémoire humaine (Atkinson et Shiffrin, 1968) qui sont la mémoire implicite (correspondant au LSTM) et la mémoire explicite. Le DM-LSTM a quatre branches extensibles selon le nombre de

modalités. Comme nous pouvons voir dans la figure 5.23, les 3 premières branches dont les émotions, la charge mentale et l'engagement sont composées de LSTMs, et la dernière branche est une simple copie des émotions passées, représentant la mémoire explicite. La mémoire explicite produit un vecteur de taille 7 représentant le nombre de classes à prédire et donc les 7 émotions de base. Le modèle proposé est capable de prédire à la fois l'émotion qui se manifeste avec une intensité maximale associée, et l'intensité de l'ensemble des 7 émotions après un temps t (spécifié à la main) à partir des données multimodales des usagers. Pour cette première expérience, la règle définie dans la mémoire explicite est la suivante : les émotions futures qui seront ressenties au temps $t+1$, sont susceptibles d'être celles d'intensités maximales ressenties par l'utilisateur pendant le temps $t-n$ à t. Le paramètre n étant la taille de la fenêtre à considérer pour la prédiction. Elle a été fixée à 3 secondes ce qui veut dire que le modèle considère les émotions ressenties aux 3 dernières secondes et prédit celles qui seront ressenties à la 4e seconde. Chaque branche se spécialise dans l'extraction des caractéristiques latentes d'une modalité.

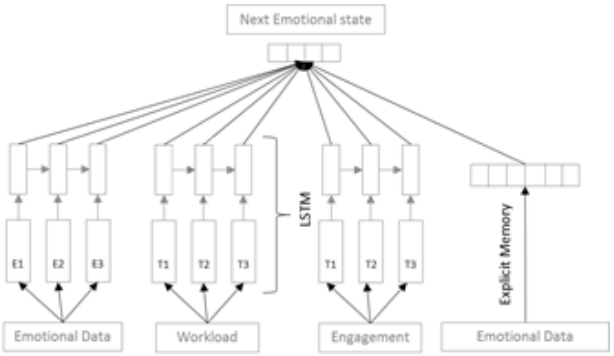


Figure 5.23 Modèle LSTM multimodal (DM-LSTM) enrichit d'une mémoire explicite pour la prédiction des émotions.

5.3.1.2 Expérimentations

Base de données de test : Seempad [8] est un ensemble de données publiques multimodales d'arguments textuels écrits en anglais, étiquetés avec des émotions provenant de l'outil *FaceReader* et des données EEG. Quatres personnes ont participé à 10 débats différents où chacun devait donner son avis sur différents sujets. Pour chaque participant, nous avions donc son état émotionnel (les 7 émotions de base), et les données EEG sur son activité cérébrale électrique (charge mentale et engagement). Il y a un total de 38764 lignes dans l'ensemble de données. Chaque ligne correspondant aux informations (les 7 émotions + Charge de travail + Engagement) enregistrées à chaque seconde.

Paramètres du modèle : Nous avons fixé la taille de la couche cachée des 3 LSTMs à 200 unités. La taille de la fenêtre a été fixée à 3 secondes (valeur arbitraire). Nous avons utilisé le *dropout* dont le pourcentage a été fixé à 75%, pour chaque couche des LSTMs. La taille du mini-lot a été fixée à 1000 et l'*epoch* a été d'abord fixée à 40 puis à 100. L'entraînement s'est fait par descente de gradient stochastique avec AAdam sur des *minis-batchs* mélangés aléatoirement. Nous avons entraîné le modèle sur 90% et 70% des données et utilisé le reste comme données de test.

Résultats et discussions Les résultats de notre modèle par rapport au LSTM standard (les données ont été concaténées et envoyées dans un LSTM unique) sur 90% et 70 % des données pour l'entraînement, sont présentés dans les figures 5.24 et 5.25. À première vue, nous pouvons déjà constater dans les 2 cas, que le DM-LTM obtient une meilleure performance par rapport au LSTM. Nous avons obtenu un maximum de 69% de précision sur l'ensemble de tests contre 67.2% pour le LSTM. Une chose intéressante que nous avons également remarquée au cours de

8. https ://project.inria.fr/seempad/datasets/

nos expériences est que la fusion au niveau de la décision mène à une meilleure performance qu'une fusion au niveau des caractéristiques comme dans le modèle LSTM ici. Par conséquent, le fait de considérer chaque modalité seule donne de

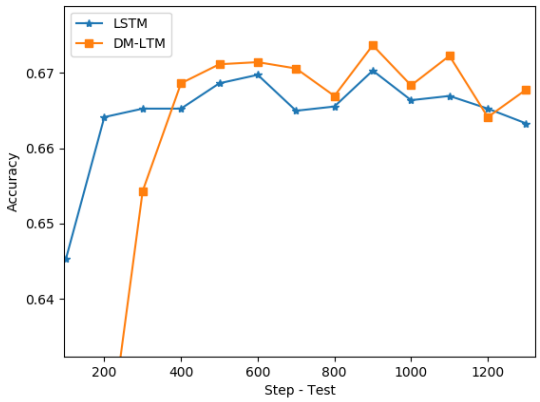


Figure 5.24 Évolution de l'*accuracy* du modèle LSTM et du modèle proposé sur les données de tests représentant 10% des données (90% étant utilisé pour l'entraînement).

meilleurs résultats dans la prédiction des émotions, comparé à la fusion brute. Cette solution est cependant un peu plus lente pendant l'apprentissage. Ceci est principalement dû à la complexité de l'architecture, ce qui signifie plus de paramètres à apprendre.

5.3.2 Détection automatique de la polarité des arguments

Les résultats présentées dans cette section ont fait l'objet d'une publication (Tato *et al.*, 2018a). Ici, nous nous sommes intéressés à la prédiction de la polarité des arguments émis par des participants durant des débats. Pour cela nous avons développé une architecture multimodale qui prend en compte plusieurs attributs de l'utilisateur incluant ses émotions, et qui est capable d'extraire une représentation latente de celui-ci, servant à la prédiction finale. L'architecture proposée

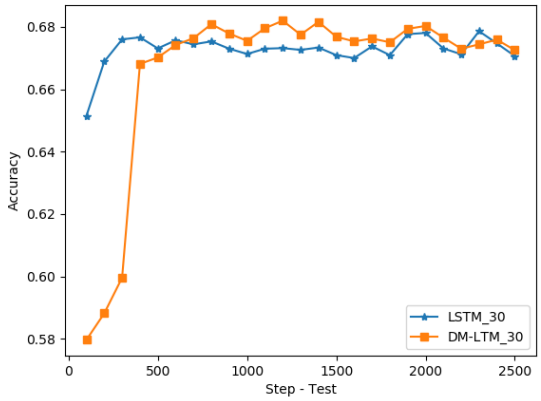


Figure 5.25 Évolution de l'*accuracy* du modèle LSTM et du modèle proposé sur les données de tests représentant 30% des données (70% étant utilisé pour l'entraînement).

consiste en la combinaison d'un LSTM, d'un CNN et de multiples réseaux de neurones *feed-forward*. Grâce à la prise en compte de la multimodalité des données et de l'intégration des informations sur les utilisateurs, l'architecture obtient de meilleurs résultats comparée à des solutions de l'état de l'art pour une tâche similaire : la détection de la polarité d'arguments.

5.3.2.1 L'architecture multimodale pour la détection de la polarité

L'architecture proposée (Us-DMN —*User Sensitive Deep Multimodal Network*—) est un réseau de neurones multimodal constitué de plusieurs branches, chacune correspondant à une modalité particulière. L'idée est de combiner les caractéristiques extraites de chaque canal d'entrée en un «vecteur commun» appelé vecteur de caractéristiques utilisateurs (l'avant-dernière couche de l'architecture proposée). Ainsi, le Us-DMN se construit dans son avant-dernière couche, une représentation latente de l'usager qui sert à la détection automatique de la polarité de l'opinion émise par cet usager. Ce modèle a été testé sur une base de données multimodale publique similaire à celle décrite dans la section précédente. Ces don-

nées proviennent de déroulement de débats où les états émotionnels, les données EEG (électroencéphalographie) et les opinions textuelles des participants ont été collectés. L'Us-DMN est construit à partir de trois architectures d'apprentissage profond : le CNN pour sa capacité à extraire des caractéristiques spatiales utiles, le LSTM pour sa capacité de modélisation temporelle et le DNN pour mapper des caractéristiques dans un espace séparable.

Dans son état actuel, le Us-DMN a 4 branches extensibles selon le nombre de modalités. Comme le montre la figure 5.26, la branche supérieure est composée d'un LSTM suivi d'un CNN pour les données ayant une structure séquentielle (dans notre cas le texte), et les 3 autres branches sont composées de DNNs pour respectivement les émotions, les données extraites du casque EEG et les autres données (ex : objet du débat, identifiant du participant, etc). Chaque branche se spécialise dans l'extraction des caractéristiques latentes pour chaque modalité. Les couches entièrement connectées sont l'avant-dernière et la dernière couche. L'avant-dernière couche est la couche représentant le vecteur de caractéristiques de l'usager, qui est considéré comme la représentation latente de ce dernier. La dernière couche est la couche correspondant aux classes et le nombre de neurones de cette couche dépend du nombre de classes à prédire. Toutes les informations nécessaires à la discrimination sont stockées dans le vecteur latent. Le vecteur appris peut être utilisé pour une autre tâche telle que la prédiction du comportement. Il informe sur l'état général actuel des utilisateurs, ce qui peut être utile pour l'adaptation. Cependant, il n'est pas interprétable du point de vue humain.

5.3.2.2 Expérimentations, résultats et discussions

Base de données : Seempad (dont la description se trouve à la section 5.3.1.2) est la base données utilisée pour les tests. C'est la même base de données que celle du projet précédent portant sur la prédiction des émotions.

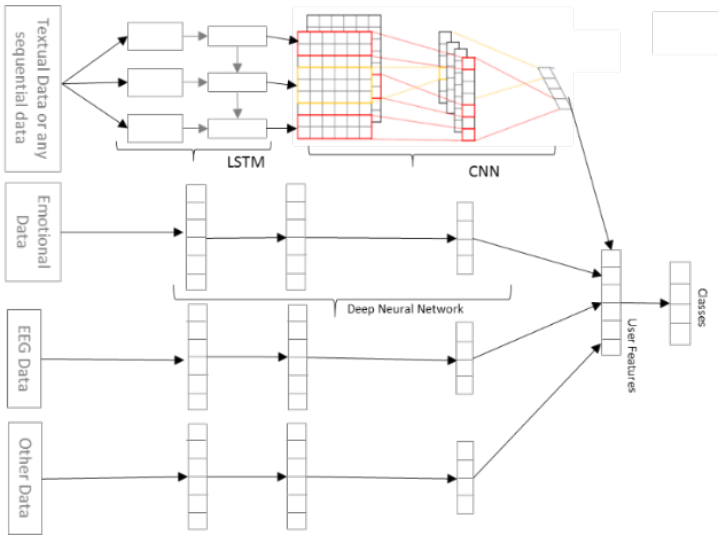


Figure 5.26 Modèle multimodal sensible aux données utilisateurs, pour la prédiction de la polarité des opinions et pour la modélisation latente de l'usager.

Paramètres du modèle : Les données textuelles (opinions) ont été représentées à l'aide de GloVe [9] (*Global Vectors for Word Representation*), un ensemble de vecteurs de mots pré-entraînés de taille 300. Les vecteurs des mots non présents dans cet ensemble ont été initialisés de façon aléatoire. Nous avons fixé la profondeur de la branche des données émotionnelles à deux et celle des branches de l'EEG et d'autres données à un, en raison de la petite taille du vecteur des données EEG qui est de deux (engagement et charge mentale), ainsi que d'autres données (participant et débat). Comme le CNN est une version légèrement modifiée de celle proposée par Kim (Kim, 2014), certains paramètres ont été choisis en fonction de leurs résultats. Le taux d'abandon (*dropout*) a été fixé à 0.5, la taille du mini-lot est de 42 et nous avons utilisé 200 *feature maps* pour chaque filtre dans le CNN. L'entraînement s'est fait avec AAdam sur des minis-batchs mélangés aléatoirement. Nous avons utilisé des filtres de petite taille (3,4,5,6) car notre ensemble de

9. https ://nlp.stanford.edu/projects/glove/

données ne contient pas de phrases très longues. La taille du vecteur latent a été fixée à 6 (déterminé par méthode de *grid-search*). Nous avons entraîné le modèle sur 85% des données étiquetées.

Résultats et discussions : Nous avons comparé notre modèle avec les méthodes traditionnelles largement utilisées dans le même contexte (détection de la polarité des opinions) à savoir le LSTM et le CNN. Nous comparons notre solution avec 4 architectures différentes construites à partir du LSTM et du CNN. La première est uniquement un LSTM (lstm-only) qui est le même utilisé dans notre solution, où toutes les données ont été concaténées. La deuxième est un CNN (cnn-only) qui également est le même que celui dans la solution proposée, où toutes les données ont été concaténées. La troisième et la quatrième sont respectivement un CNN (cnn+user-data) et un CNN+LSTM (cnn+lstm) dont toutes les données sauf celles contenant les informations sur les utilisateurs (débat, participant, etc.) ont été concaténées avant d'être envoyés à l'architecture et que les données sur les utilisateurs ont été concaténées avec la sortie de l'architecture pour la prédiction finale. Bien que nous nous attendions à des gains de performance grâce à

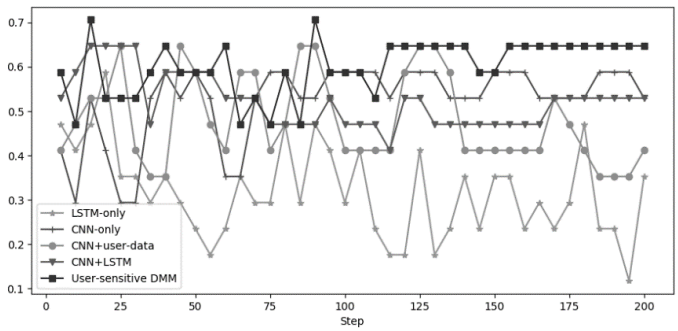


Figure 5.27 Évolution de la précision dans le temps des modèles CNN, LSTM Us-DMN sur les données de test.

l'utilisation de vecteurs pré-entraînés avec le modèle LSTM ou CNN, nous avons été surpris qu'ils ne donnent pas de bons résultats individuellement sur notre en-

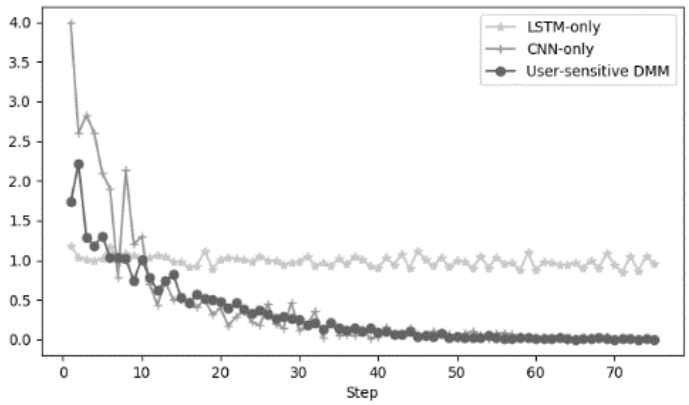


Figure 5.28 Évolution de la valeur de la fonction de perte pendant l'étape d'apprentissage.

semble de données. Les résultats de notre modèle par rapport à ces méthodes sont présentés dans les figures 5.27, 5.28 et 5.29. À première vue, nous pouvons constater que le US-DMN a surpassé toutes les autres techniques vers la fin de l'entraînement. Nous avons obtenu 71% de précision contre 64%, 64%, 58% et 47% respectivement pour les modèles CNN+LSTM, CNN+User-data, CNN only et LSTM only lors des tests (figure 5.24). Une chose intéressante que nous avons remarquée sur ces résultats est que les modèles CNN+LSTM et CNN+User ont pratiquement la même performance sauf que le premier prend plus de temps pour l'entraînement que la seconde. Par conséquent, la prédiction des comportements des utilisateurs dépend non seulement des données utilisateur mais aussi de l'architecture utilisée pour extraire une représentation des utilisateurs.

Notre solution est cependant un peu plus lente dans le processus d'entraînement comparé aux autres techniques. Ceci est principalement dû à la complexité de l'architecture, ce qui signifie plus de paramètres à entraîner.

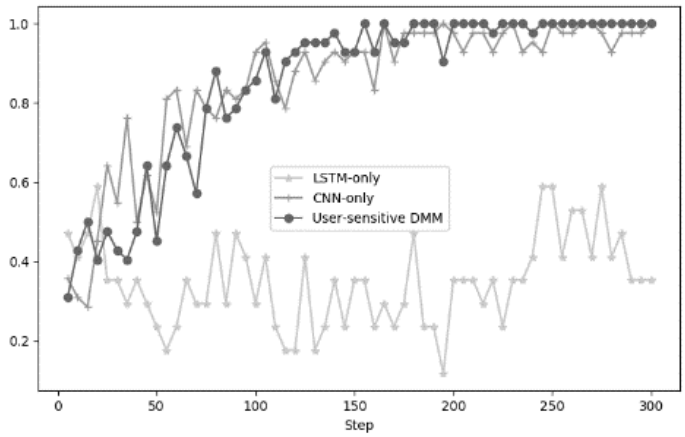


Figure 5.29 Évolution de la valeur de la précision pendant l'étape d'entraînement.

5.3.3 Prédiction de l'attention et de la charge de travail durant la résolution de problèmes mathématiques

Les résultats présentés dans cette section ont fait l'objet d'une publication (Tato *et al.*, 2019b). Il a été démontré que l'attention et la charge de travail cognitive peuvent prédire la performance des élèves lorsqu'ils résolvent des problèmes (Ghali *et al.*, 2018). Ainsi, si nous sommes capables de prédire l'attention et la charge de travail d'un apprenant en temps réel pendant la résolution de problèmes, nous serons en mesure de prédire ses chances de réussite. Alors que les études existantes visent à classer les états mentaux en temps réel, leur prédiction n'a pas encore été explorée. Par conséquent, nous avons mené une étude durant laquelle nous avons mis au point un LSTM utilisant notre concept d'intégration de connaissances a priori et a posteriori dans les réseaux de neurones en utilisant le mécanisme d'attention, pour prédire l'attention humaine de l'élève et sa charge cognitive dans la résolution de problèmes. Ces deux états mentaux sont extraits des signaux EEG. L'objectif est d'arriver à un modèle capable de prédire avec précision ce que serait l'état mental de l'apprenant au temps $t+1$ sachant ce qu'il en était au temps

$t, ..., t-T$ où T représente le pas de temps du modèle. Un tel modèle peut être utile dans un tuteur intelligent pour aider à déduire les états mentaux de l'apprenant et décider en conséquence de ce qu'il faut faire pour optimiser l'apprentissage.

5.3.3.1 Expérimentations

La base de données de test : Des chercheurs de l'université de Montréal (Ghali *et al.*, 2018) ont mené une expérience où ils ont demandé à des élèves du primaire (4e et 5e années) de résoudre les tâches sélectionnées dans l'environnement NetMath [10]. Les tâches proposées ont été conçues pour les élèves de niveau supérieur (6e année). Dix sept apprenants (10 F, 7 M) ont participé volontairement à cette étude. Ces apprenants étaient âgés de 9 à 11 ans (Moyenne = 10.05 ; Ecart type = 0.42). Un total de dix tâches de cette plate-forme a été sélectionné. Ces tâches sont divisées en trois niveaux de difficulté : facile, moyen et difficile. Pendant l'exécution des tâches, des données en temps réel ont été collectées à partir du casque EEG *Neeuro Senzeband*. Ce casque a permis d'obtenir des données brutes d'EEG à partir de quatre canaux et de trois mesures d'état mental (attention, charge de travail et relaxation).

Chacun des 17 participants a passé environ 30 minutes à résoudre tous les problèmes. Pour chacun de ces participants, les données collectées sont : l'attention, la charge de travail, la relaxation, et l'activité des deux hémisphères du cerveau gauche/droite. Nous nous sommes concentrés uniquement sur les trois premières variables. Pour la prédiction de l'attention et de la charge de travail, nous avons défini une fenêtre de temps de 20 secondes (sachant les réactions collectées durant les 20 dernières secondes, quelles seront les réactions à la 21eme seconde ?). L'ensemble de données a été divisé en données d'entrée X et données de sortie Y où chaque ligne de X représente des événements survenus 20 secondes auparavant et

10. https ://www.netmath.ca/fr-qc/

Time	Attention	Workload	Relaxation	Left	CenterLeft	CenterRight	Right
16:13:29 PM	0,998691	0,3732375	0,1199541	-6	-13	-8	-2
16:13:30 PM	0,01	0,01	0,07488824	8	10	68	15
16:13:32 PM	0,89289	0,01	0,01517821	-14	11	113	29
16:13:32 PM	0,89692	0,01	0,05044472	-6	-7	6	10
16:13:33 PM	0,904556	-0,07081398	0,07155754	15	-6	-46	-5
16:13:34 PM	1	0,2673379	0,06456525	13	11	-9	9
16:13:35 PM	0,968494	0,01	0,0776386	8	4	-2	6
16:13:36 PM	0,969414	0,5121103	0,09333985	50	9	9	16
16:13:37 PM	0,359994	0,01	0,0469471	47	-2	-21	9
16:13:38 PM	0,724037	0,276055	0,09904281	39	4	3	13
16:13:39 PM	0,970849	0,333482	0,1598548	19	3	6	9

Figure 5.30 Extrait des données brutes de l'EEG.

chaque ligne de Y est une étiquette qui est l'événement survenu à la 21e seconde. Cette configuration nous a donné environ 20000 lignes de données dont 75 % ont été utilisées pour l'entraînement et 25% pour les tests. La Fig. 5.30 montre un extrait de la base de données.

L'architecture proposée : Nous avons d'abord entraîné un modèle de régression (LSTM avec erreur quadratique moyenne) dont l'objectif était de prédire la valeur approximative de l'attention/charge de travail mais ce premier modèle nous a donné de mauvais résultats. Nous avons donc transformé le problème en un problème de classification où nous prédisons si la valeur de l'attention est supérieure ou non à une valeur seuil. La meilleure valeur seuil trouvée était de 0.6 (0.02 pour la charge de travail). Le modèle conçu est donc capable de prédire si l'attention sera élevée (> 0.6) ou faible (≤ 0.6). Par exemple, si la valeur à prédire est inférieure à 0.6, alors l'étiquette est codée comme étant $[1, 0]$ sinon l'étiquette est codée comme étant $[0, 1]$. Le modèle (voir Fig. 5.31) est composé d'une couche d'entrée contenant 3 neurones représentant chacune les 3 variables sélectionnées. Les valeurs brutes ont été retenues. Après la couche d'entrée, il y a deux couches LSTM capables d'extraire les informations séquentielles utiles à la prédiction. Des informations contextuelles ont également été ajoutées au modèle (deuxième branche) via le mécanisme d'attention : informations sur la difficulté

du problème (3 neurones) et le niveau de l'élève (1 neurone où la valeur est 0 lorsque sa performance en classe est inférieure à la moyenne, 1 sinon). L'avant-dernière couche résume l'information extraite dans les couches inférieures afin de l'envoyer à la couche finale qui fait la prédiction finale. La fonction d'activation *ReLU* (*Rectified Linear Unit*) est utilisée dans toutes les couches sauf la dernière où la fonction d'activation est le *Softmax*. La perte est l'entropie binaire croisée.

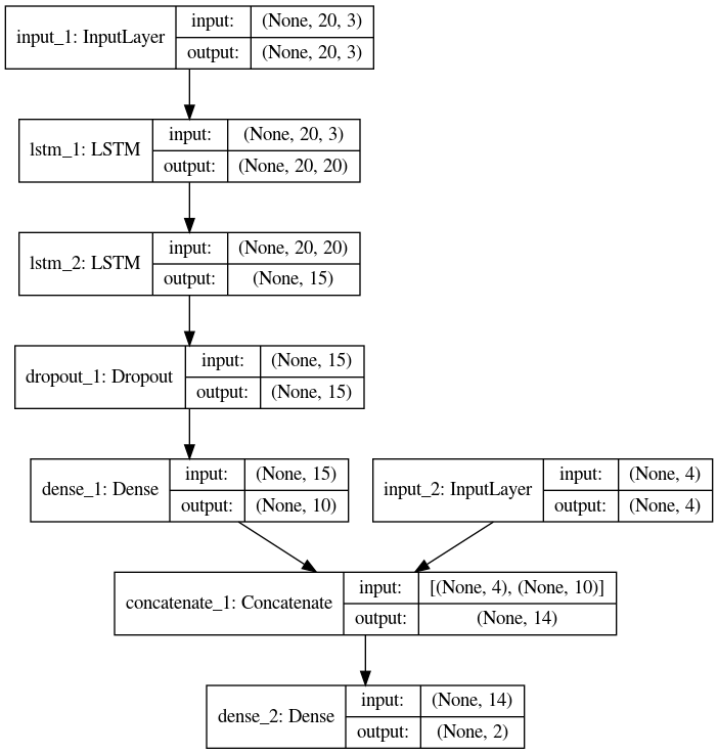


Figure 5.31 Architecture pour la prédiction de la variation en temps réel de l'attention et de la charge de travail.

Résultats et discussions : Le modèle qui prédit l'attention a donné 67% de précision et le modèle qui prédit la charge de travail a donné 79% de précision, ce qui est la moyenne sur 20 entraînements différents. Ces résultats montrent qu'il est possible d'observer la variation de l'attention et de la charge de travail des élèves pendant la résolution de problèmes et de prédire comment ces états mentaux varieront en temps réel. Cependant, nous remarquons que le premier

modèle est moins performant que le deuxième, ce qui signifie qu'il est difficile de prédire comment l'attention variera dans le temps par rapport à la charge de travail. Le modèle tel qu'il est conçu est un modèle général en ce sens qu'il ne tient pas compte des spécificités de chaque personne. Cependant, comme chaque personne est différente et peut avoir des comportements différents des autres sur la même tâche, il est important d'intégrer plus de données contextuelles telles que le style d'apprentissage ou la personnalité. Néanmoins, nos résultats semblent prometteurs. Les détecteurs automatiques d'état mentaux basés sur l'EEG devront probablement être beaucoup plus précis en premier lieu pour aider à la conception d'un prédicteur précis et pour aider les tuteurs intelligents en temps réel.

5.4 Disponibilité des solutions et publications

La majorité de nos solutions sont disponibles sous licence libre sur notre répertoire GitHub [11] pour favoriser les recherches dans ce domaine. Cependant, certaines des propositions ne sont pas disponibles puisque associé à des projets dont nous n'avons pas suffisamment les droits pour rendre publiques les solutions. Dans ce répertoire, on y retrouve principalement :

- Notre solution d'optimisation (en plusieurs versions) codée avec Python. Deux différents projets ont été créés : le premier projet contient le code utilisable sous *Keras* et l'autre utilisable sous *Tensorflow* . Nous y incluons également les tests effectués.
- Le DKTm augmenté de l'attention pour le traçage des connaissances en raisonnement logique. On y retrouve également un test effectué sur les données publiques *Assistments*. *Assistments* est un tuteur en ligne qui enseigne et évalue simultanément les mathématiques aux élèves du primaire.

11. https ://github.com/angetato/

Il s'agit, à notre connaissance, de l'ensemble de données de traçage des connaissances, le plus important accessible au public.

- Le modèle permettant la prédiction du niveau de raisonnement en utilisant l'attention et les connaissances a priori/ a posteriori. Ici les données ne sont pas disponibles.

Ces solutions peuvent être réutilisées et ou modifiées sous réserve de citer cette thèse ou les articles publiées (voir la sous section 5.4.1) dans le cadre de cette recherche.

5.4.1 Publications

Ici, nous présentons par ordre chronologique les publications acceptées et publiées, liées à cette thèse.

1. A.Tato, R. Nkambou. Accelerating First-Order Optimization Algorithms. Accepté (Student abstract AAAI 2020).
2. A.Tato, R. Nkambou and A. Dufresne. Using AI techniques in a Serious Game for Socio-moral Reasoning Development. Accepté (EAAI 2020).
3. A.Tato and R. Nkambou. Deep Knowledge Tracing On Limited Data. Accepted (IEEE ICTAI 2019).
4. A.Tato and R. Nkambou. Accelerating First-Order Optimization Algorithms. Accepted (Springer ICONIP 2019).
5. A. Tato, R. Nkambou and A. Dufresne (2019). Hybrid Deep Neural Networks to Predict sociomoral Reasoning skills. Proceedings of the 12th International Conference on Educational Data Mining (EDM'19). pp. 623-626.
6. A. Tato, R. Nkambou and R. Ghali (2019). Towards Predicting Attention and Workload During Math Problem Solving. Proceedings of the 15th International Conference on Intelligent Tutoring Systems (ITS'19), Springer,

pp. 224-229.

7. Mercier, J., Chalfoun, P., Martin, M., Tato, A. A., and Rivas, D. (2019). Predicting Subjective Enjoyment of Aspects of a Videogame from Psychophysiological Measures of Arousal and Valence. Proceedings of the 15th International Conference on Intelligent Tutoring Systems (ITS'19), Springer, pp. 174-179).
8. A. Tato, R. Nkambou , A. Dufresne, and C. Frasson (2018). Semi-Supervised Multimodal Deep Learning Model for Polarity Detection in Arguments. In 2018 International Joint Conference on Neural Networks (IJCNN), IEEE, pp. 1-8.
9. A. Tato, R. Nkambou and C. Frasson (2018). Predicting Emotions From Multimodal Users' Data. In Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization (UMAP'18). ACM, pp. 369-370.
10. A. Tato, A. Dufresne and R. Nkambou. Preliminary Evaluation of a Serious Game for sociomoral Reasoning. Proceedings of the 14th International Conference on Intelligent Tutoring Systems (ITS'18), LNCS 10858, Springer, pp. 473–475.
11. A.A. Nyamen Tato, R. Nkambou and A. Dufresne (2017). Convolutional Neural Network for Automatic Detection of Sociomoral Reasoning Level. In : X. Hu, T. Barnes, A. Hershkovitz, L. Paquette (Eds), Proceedings of the 10th International Conference on Educational Data Mining (EDM'17), pp. 284-289.
12. Tato, A., Nkambou, R., Brisson, J., and Robert, S. (2017, June). Predicting Learner's Deductive Reasoning Skills Using a Bayesian Network. In International Conference on Artificial Intelligence in Education, LNCS 1033, (pp. 381-392). Springer.
13. Tato, A., Nkambou, R., Brisson, J., Kenfack, C., Robert, S., and Kissok,

P. (2016, September). A Bayesian Network for the Cognitive Diagnosis of Deductive Reasoning. In European Conference on Technology Enhanced Learning (pp. 627-631). Springer International Publishing.

5.5 Conclusion

Dans ce chapitre, nous avons passé en revue les différentes applications dans lesquelles nos solutions ont été testées avec succès. Nous avons également présenté l'ensemble des publications faites tout au long de cette recherche. Comme nous l'avons déjà mentionné, les solutions que nous avons présentées sont disponibles en licence libre sous réserve de citer cette recherche. Le prochain chapitre conclut ainsi cette thèse.

CONCLUSION

Rappel de la problématique

L'apprentissage profond grâce aux différentes architectures qui l'implémentent, permet à partir des données brutes, d'extraire efficacement des informations cachées importantes pour la discrimination et la modélisation des données présentées en entrées. Après avoir montré en quoi il était essentiel de tendre vers ce type de solution pour la modélisation de l'usager, nous avons mis en évidence le retard accumulé par les solutions actuelles ainsi que leurs lacunes. Ceci étant principalement dû à des approches de développement figées, basées sur des théories très peu évolutives, des techniques moins performantes et moins appropriées à la modélisation du comportement holistique de l'usager qui est multimodale, et surtout non adaptée dans un contexte où les données sont de plus en plus disponibles. Les rares propositions faisant usage des techniques d'apprentissage profond ignorent les connaissances existantes, ou ne tiennent pas compte du balancement des données (connaissances rares), éléments essentiels sur lesquels miser pour une meilleure modélisation. Par ailleurs, ces techniques nécessitent parfois d'être ajustées pour qu'elles soient adaptées à un contexte où les réponses doivent être précises et fournies en temps réel (comme dans le cas des environnements interactifs d'apprentissage), puisqu'elles sont souvent coûteuses en temps. Nous avons présenté dans notre état de l'art les principaux travaux pertinents du point de vue de la modélisation de l'usager en utilisant des théories existantes mais également des techniques usuelles d'apprentissage machine pour la catégorisation des comportements dans les systèmes interactifs pour des fins d'adaptation. Nous avons également montré en quoi certaines approches constituaient une avancée dans le

domaine de la modélisation de l'usager en utilisant des techniques plus avancées comme l'apprentissage profond, bien qu'aucun modèle à ce jour ne réponde aux exigences d'une représentation efficace, multifacette, temps réel et hybride.

Résumé des contributions

Les contributions les plus pertinentes du point de vue de notre problématique sont les différentes solutions apportées aux modèles d'apprentissage profond afin de les adapter au domaine de la modélisation de l'usager. Ainsi, nous avons proposé et utilisé dans cette thèse, différentes techniques d'apprentissage profond principalement pour une modélisation plus efficace de l'usager et pour l'extraction automatique de règles d'adaptation. Pour prouver l'efficacité de ces solutions dans le cas particulier des systèmes interactifs hautement adaptatifs, nous les avons intégrés et testés dans plusieurs applications.

Notre première solution a visé l'adaptation fondamentale des techniques d'apprentissage profond à la modélisation de l'usager. Elle a eu pour but la proposition d'une solution d'accélération de l'apprentissage dans les réseaux de neurones profonds, puisqu'une modélisation de l'usager doit se faire en temps réel et doit être le plus fidèle possible. Elle a consisté à accélérer la descente de gradient pour les algorithmes d'optimisation de premier ordre (utilisant la dérivée première du gradient) et à trouver une meilleure solution. L'idée étant d'accélérer la descente du gradient lorsque d'une itération à une autre, sa direction ne change pas et vice versa. Nous avons montré que cette technique est capable d'accélérer l'apprentissage et trouver un minimum d'une valeur plus basse plus rapidement, de n'importe quel algorithme classique de l'état de l'art (Adam, Adagrad, etc.). Toutes les autres solutions proposées dans le cadre de cette thèse utilisent les versions accélérées des algorithmes de base dans le processus d'apprentissage.

Les autres solutions développées sont trois techniques (métamodèle) de modélisa-

tion de l'usager, fondées chacune sur une architecture originale qui la rend efficace dans une perspective importante de la modélisation de l'usager. Notre premier modèle permet la capitalisation des connaissances existantes (à priori/a postériori) pour améliorer la représentation de l'usager. La représentation résultante est matérialisée par un modèle prédictif d'un ou plusieurs aspects caractérisant l'usager, ex : sa connaissance, ses émotions. Cette solution utilise le mécanisme d'attention pour consulter les connaissances existantes afin d'améliorer la prédiction finale. Cette solution a été appliquée à la prédiction du niveau de raisonnement socio-moral à partir d'un justificatif textuel où, la connaissance existante n'est autre qu'un descriptif conçu par les experts et associé à chaque niveau de raisonnement. La solution a été implémentée dans un jeu sérieux adaptatif que nous avons développé et qui vise l'apprentissage du raisonnement socio-moral. Les résultats expérimentaux ont montré son efficacité pour la prédiction du niveau de raisonnement socio-moral comparé aux solutions de l'état de l'art.

Le deuxième modèle proposé vise à résoudre le problème de connaissances rares que l'on peut rencontrer dans certains systèmes visant l'apprentissage. En rappel, nous appelons connaissances rares, les connaissances difficiles à maîtriser, dont les apprenants auront tendance à donner beaucoup moins de réponses correctes (peu de données sur les apprenants ayant réussis) et les connaissances faciles à maîtriser et pour lesquelles tous les apprenants auront tendance à donner de bonnes réponses sur des problèmes portant sur cette connaissance (beaucoup de données sur les apprenants ayant réussis et peu de données sur ceux ayant échoués). Pour résoudre ce problème, nous avons suggéré une modification à la fonction de perte ou d'erreur. Cette fonction évalue la différence entre le résultat escompté et celui produit pour l'architecture et qui est utilisée par l'architecture pour s'ajuster. La modification implique d'ajouter un masque dont le but est de masquer les autres connaissances afin de donner plus de poids aux connaissances rares. Ainsi,

on force l'architecture à faire de bonne prédiction pour ces connaissances en la pénalisant davantage lorsqu'elle fait des erreurs. Nous attribuons des poids à ces connaissances dans le calcul de la fonction de perte. Cette solution a été appliquée au traçage des connaissances dans Muse-logique (un STI pour l'apprentissage du raisonnement logique) où il s'agissait de prédire les compétences en raisonnement logique sachant les réponses aux exercices. La solution implémentée a donné de meilleurs résultats en comparaison aux solutions de l'état de l'art à savoir le DKT et le BKT. De plus, le réseau bayésien qui est créé entièrement par les experts du domaine, étant une connaissance existante, a été fusionné à la solution en utilisant le mécanisme d'attention de la première solution que nous avons proposée.

Le troisième modèle proposé vise la prise en compte de la nature multimodale du comportement humain dans les architectures profondes. La multimodalité du comportement humain est un aspect qui est le plus souvent négligé dans la modélisation de l'usager pourtant nécessaire pour une modélisation plus fidèle. Ici, nous supposons avoir accès à au moins 2 modalités de l'usager (ex : ses émotions, son état cognitif, etc.). L'idée est d'extraire des informations pertinentes de chaque modalité dans un premier temps (*decision level fusion*), ensuite d'en extraire d'autres à partir de la fusion des informations extraites de chaque modalité. Ainsi les informations pertinentes pour la prédiction et la modélisation du comportement sont d'abord extraites séparément, puis sont combinées avec celles provenant d'une extraction fusionnée. Cette solution a été appliquée avec succès à la prédiction des émotions et à la prédiction de la polarité des arguments.

En plus de ces solutions de modélisation de l'usager, nous avons proposé une solution d'extraction automatique de règles d'adaptation. Cette solution utilise principalement les arbres de décision et les réseaux de neurones. Elle prend les données d'interactions en entrées et en ressort avec des résultats permettant d'apporter des modifications au système interactif dans le but de répondre aux besoins

des usagers. Elle a été testée dans le jeu sérieux *LesDilemmes* (visant le développement des compétences en raisonnement socio-moral) développé par notre équipe, dont le but était d'évaluer subjectivement et objectivement son impact sur les usagers. Nous avons comparé le jeu dans sa version non adaptative c'est-à-dire ne contenant aucune de nos solutions, à sa version adaptative où certaines de nos solutions ont été incorporées dont le modèle de l'usager hybride pour la prédiction du raisonnement socio-moral et les règles d'adaptation extraites automatiquement par apprentissage machine. Nous avons pu constater que la version adaptative a eu un impact positif et significatif sur le gain d'apprentissage, les émotions et la satisfaction par rapport à sa version non adaptative. Cependant, ce gain d'apprentissage pourrait être meilleur si la transcription audio vers texte est améliorée. Nous avons vu que la transcription automatique du justificatif verbal de l'audio vers le texte a un impact négatif sur notre système de cotation automatique, ce qui biaise l'adaptation dans le jeu vu que la connaissance n'est pas très bien évaluée en temps réel. Une solution sur laquelle nous travaillons actuellement est de développer une version en anglais du jeu puisque la transcription automatique de l'audio vers le texte est plus avancée pour l'anglais que pour le français. Cependant, ceci impliquera de ré-entraîner notre modèle de cotation automatique sur des verbatims en anglais qui devront être au préalable recueillis et annotés par les experts.

Les techniques et architectures que nous avons présentées dans cette thèse sont exploitables dans des domaines autres que l'éducation. La technique d'accélération des algorithmes d'optimisation dans les réseaux de neurones peut être appliquée à n'importe quel problème l'apprentissage impliquant une optimisation de premier ordre par descente de gradient. L'architecture hybride quant à elle, peut être utilisée dans les contextes où des connaissances a priori et a posteriori sont à portée de main et que l'on veut augmenter la capacité de prédiction des modèles développés.

En plus cette méthode permet de pallier dans une certaine mesure les problèmes de données déséquilibrées et les problèmes où il n'y a pas assez de données (peu de données pour l'apprentissage d'où une mauvaise généralisation). Elle permet donc de concevoir des modèles ayant un plus fort taux de généralisation qu'un modèle standard. Nous avons proposé une amélioration importante de l'algorithme DKT. Puisque ce dernier tel que présenté dans la littérature ne tient pas compte des connaissances rares pouvant survenir dans n'importe quel contexte éducationnel. Nous avons donc, grâce à une technique simple de pénalisation de la fonction d'erreur, pu améliorer le DKT dans ce sens. Nous avons proposé une architecture multimodale qui combine plusieurs modalités pas seulement en les concaténant mais en apprenant également des *features* qui peuvent coexister entre les différentes modalités. Nous avons également proposé des techniques pour concevoir un modèle d'adaptation qui utilise des règles d'adaptation extraites automatiquement à partir de réseaux de neurones et d'arbres de décision.

Le premier avantage de nos solutions est qu'elles sont facilement adaptables à n'importe quel contexte de modélisation et même dans des contextes ne visant pas forcément la modélisation de l'usager. Les solutions conventionnelles ne sont pas souvent transposables dans d'autres domaines autres que ceux dans lesquels ils ont été testés. À l'inverse, en plus de résoudre des problèmes concrets rencontrés dans les solutions de l'état de l'art, nos solutions permettent de faciliter le développement et la mise en place d'une modélisation de l'usager pour une adaptation efficace quel que soit le domaine, comme nous l'avons vu dans les différents contextes applicatifs testés. Le méta-modèle proposé est entièrement modulable, et le code source disponible. L'ensemble des objectifs visés dans le cadre de cette recherche, notamment le développement de techniques pour permettre une modélisation efficace de l'usager dans un premier temps et pour une adaptation effective ont été atteints. Cependant, dans la perspective d'un développement futur, des

ajouts ou améliorations pourraient être envisagés tant au niveau conceptuel qu'au niveau de l'implémentation des solutions.

Les résultats de cette recherche contribuent à l'avancement des connaissances tant au niveau fondamental qu'au niveau applicatif. Au niveau fondamental, une nouvelle technique d'accélération de l'apprentissage dans les réseaux de neurones ainsi que différentes approches d'amélioration des résultats de modélisation des usagers par apprentissage profond dont la prise en compte de la mulimodalité du comportement humain ont été proposés. Déterminer et considérer un profil social du joueur-apprenant en se basant sur son niveau de raisonnement socio-moral, est d'une part, une opportunité d'offrir une interaction sociale plus riche dans un système interactif et, d'autre part, une dimension innovante qui fera certainement avancer la recherche dans le domaine de la socialisation dans les systèmes informatiques intelligents. L'adaptation se basant sur notre méta-modèle de l'usager enrichi donne lieu à une optimisation de l'expérience de jeu (le plaisir), et offre un cadre plus motivant pour le développement des compétences visées comme nous avons pu le constater. Les règles générées par le modèle adaptatif ouvrent des portes à l'exploration de nouvelles techniques pédagogiques efficaces et de nouveaux styles d'apprentissage. D'un point de vue applicatif, le modèle d'adaptation proposé ainsi que les méthodes et outils sous-jacents permettront d'outiller les développeurs de systèmes interactifs dans la création d'une nouvelle génération de systèmes cognitivement, émotionnellement et socialement plus informés, plus adaptatifs et plus intelligents. Par ailleurs, nous visons une réutilisabilité de nos solutions dans des domaines où l'adaptation et la prédiction du comportement de l'usager est un enjeu. Parmi des exemples de domaines potentiels qui pourront en bénéficier, nous pouvons citer, les outils de formations, l'apprentissage en ligne impliquant les MOOCs (Massive Open Online Course), les jeux vidéo, etc.

Validation des hypothèses

Hypothèse 1 : *Les recherches dans le domaine de l'intelligence artificielle en éducation ont montré que l'intégration d'un modèle sophistiqué de l'utilisateur dans les environnements d'apprentissage, entraînerait une adaptation plus flexible et efficace. Nous faisons l'hypothèse que ce constat est applicable dans tous systèmes interactifs d'apprentissage (jeux sérieux par exemple).*

Hypothèse 2 : *Un modèle enrichi de l'usager qui inclut différents facteurs explicites (lorsque disponibles et captés par différentes modalités) liés aux compétences cognitives et méta-cognitives de celui-ci, à son profil affectif, à sa personnalité et son profil social, permettra une adaptation réussie en terme d'efficacité pédagogique.*

Ces deux hypothèses ont pu être validées pendant les expérimentations avec la version adaptative du jeu *LesDilemmes*, où le modèle du joueur était constitué de son état connaissances et de son état émotionnel. Nous avons pu constater que l'adaptation basée sur ce modèle riche du joueur a permis un gain significatif de l'apprentissage, comparé à la version du jeu ne contenant aucun modèle.

Hypothèse 3 : *Les architectures profondes ont la capacité d'extraire des informations latentes permettant d'abstraire assez fidèlement les données fournies en entrées permettant ainsi de les prédire et ou de les classifier. Nous faisons l'hypothèse qu'en nourrissant ces architectures de données d'utilisateurs (ex : apprenants), il sera possible d'extraire des représentations latentes et assez précise de ces derniers.*

Hypothèse 4 : *Le développement et/ou l'amélioration des techniques de fouille de données et d'apprentissage machine existantes permettront une analyse efficace des données d'interaction multimodales et multi-sources (historique d'actions, expressions faciales, regard, verbatim, etc.).*

Ces deux hypothèses ont également pu être validées lors de la prédiction des connaissances dans le STI Muse-logique, et de la prédiction des émotions, de l'attention, de la charge mentale et de la polarité des arguments dans divers contextes. Cependant, la représentation latente de l'usager qui en est extraite n'est pas interprétable et n'a donc pas pu être utilisée dans le cadre de cette recherche.

Perspectives futures

Même si les solutions proposées ont été testées et validées dans différents contextes applicatifs, il reste toujours plusieurs aspects à améliorer, et elles peuvent être affinées et généralisées. Une validation exhaustive des différents modèles nécessiterait notamment des bases de données beaucoup plus volumineuses ainsi que des systèmes interactifs déjà en déploiement. Nos approches pour le problème de modélisation holistique de l'usager ouvrent par ailleurs de nombreuses perspectives d'amélioration et d'extension. Nous avons par exemple évoqué des problèmes de connaissances rares qui n'étaient jusqu'alors pas pris en compte. Nous avons également évoqué le problème de temps réel des approches d'apprentissage profond, qui prennent souvent un temps non négligeable pour l'apprentissage d'une représentation alors que la modélisation de l'usager doit se faire le plus vite possible pour que le système ait le temps d'y répondre ; de plus la représentation qui en sort doit être suffisamment précise pour assurer une adaptation pertinente. Une perspective peut-être lointaine mais des plus intéressantes serait une généralisation de nos solutions aux systèmes interactifs plus généraux, dans le but de tendre vers une nouvelle génération de systèmes interactifs adaptatifs plus attentifs aux besoins de ses usagers.

Une question peut se poser sur les métriques utilisées pour l'évaluation d'un modèle de l'usager. Pour l'instant, nous avons utilisé la précision pour quantifier l'efficacité (la fidélité) de nos modèles. Est-ce une bonne métrique pour ce problème ?

Comment mesure-t-on la fidélité d'un modèle par rapport à l'usager modélisé ? Est ce que prédire avec précision les actions d'une personne implique qu'elle peut être remplacé par le prédicteur ? Bien entendu, la mesure du gain d'apprentissage comme nous l'avons fait, est une métrique sans équivoque pour évaluer l'efficacité et la précision d'un modèle usager. Cependant, cette métrique implique de conduire des expérimentations sur le terrain, tâche qui est très coûteuse en temps. Pourrait-on penser à une métrique spécifique à l'évaluation de la «fidélité» d'une modélisation sans toutefois évaluer le système interactif au complet ? Des travaux futurs dans cette perspective pourraient s'inspirer des travaux sur les réseaux de neurones antagonistes (*adversarial networks*), où l'idée est de voir à quel point le réseau principal (qui modélise l'usager) est capable de différencier cet usager parmi d'autres, sachant que le réseau adverse est celui qui génère des «fausses» interactions.

Certaines de nos solutions sont disponibles sur *Github* en licence libre. Toutes les solutions ont été implémentées avec python sauf les environnements qui ont été implémentés avec *Unity3D* (C# pour le jeu sérieux) et *JEE*. Nous espérons que ces feront avancer les recherches et seront d'une grande aide pour les concepteurs de modèle usager, de système adaptatif mais également pour les praticiens de l'apprentissage profond. De manière générale, les résultats que nous proposons pourrons probablement contribuer à l'avancement des connaissances dans les domaines suivants : apprentissage profond, jeux vidéo, systèmes tutoriels intelligents, Intelligence artificielle en éducation ainsi que dans la modélisation des usagers, l'adaptation et la personnalisation.

APPENDICE A

PREUVE DU THÉORÈME 3.3.3

La preuve présentée ci-dessous s'inspire du théorème 4 dans (Reddi *et al.*, 2018), qui fournit une preuve de convergence pour AMSGrad.

Le but est de prouver que, le *regret* de AAMSGrad est délimité par :

$$\begin{aligned} R_T \leq & \frac{D_\infty^2}{2(1-\beta_1)} \sum_{i=1}^{d} (\frac{\hat{v}_{T,i}^{-1/2}}{\alpha_T}) + \frac{D_\infty^2}{4(1-\beta_1)} \sum_{t=1}^{T} \sum_{i=1}^{d} \frac{\beta_{1t} \hat{v}_{t,i}^{1/2}}{\alpha_t} \\ & + \frac{2\alpha\sqrt{1+log(T)}}{(1-\beta_1)^2(1-\gamma)\sqrt{(1-\beta_2)}} \sum_{i=1}^{d} \|g_{1:T,i}\|_2 \end{aligned} \tag{A.1}$$

Démonstration. Pour chaque méthode d'optimisation, nous avons :

$$x_{t+1} = x_t - \alpha U \tag{A.2}$$

Où α est la taille du pas. Notez que la valeur de la mise à jour U est le gradient (ou pente) d'une ligne. Par exemple, cette ligne représente la pente de la tangente de la perte au point x_t dans le cas du SGD.

AAMSGrad a 2 règles pour la mise à jour, qui sont différentes selon que la direction de la mise à jour change ou non. Le calcul du point x_{t+1} est effectué comme suit :

$$
\begin{aligned}
\text{(a) } x_{t+1} &= \Pi_{F,\sqrt{\hat{V}_t}}(x_t - \alpha_t m_x / \sqrt{\hat{v}_t}) \qquad \text{si } g_t \cdot m_{t-1} > 0 \\
&\leq \Pi_{F,\sqrt{\hat{V}_t}}(x_t - 2\alpha_t \cdot |m_t| / \sqrt{\hat{v}_t}) \\
m_x &= \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot (\nabla_{\theta_t} J(\theta) + m_{t-1}) \\
\text{(b) } x_{t+1} &= \Pi_{F,\sqrt{\hat{V}_t}}(x_t - \alpha_t m_t / \sqrt{\hat{v}_t}) \qquad \text{sinon}
\end{aligned}
\tag{A.3}
$$

L'inégalité en (a) est due au fait que, puisque $g_t \cdot m_{t-1} > 0$ et $\beta_1 = 0.9$, nous pouvons écrire :

$$
|(1-\beta_1)m_{t-1}| \leq |\beta_1 m_{t-1} + (1-\beta_1)g_t| \tag{A.4}
$$

Ce qui nous conduit à :

$$
\begin{aligned}
m_x &= \beta_1 m_{t-1} + (1-\beta_1)g_t + (1-\beta_1)m_{t-1} \\
m_x &\leq |2 \cdot (\beta_1 m_{t-1} + (1-\beta_1)g_t)| \\
m_x &\leq |2 \cdot m_t|
\end{aligned}
\tag{A.5}
$$

Toutes les opérations sont élément par élément. Lorsque la direction de la mise à jour actuelle change par rapport à la précédente, la mise à jour actuelle est la même que dans AMSGrad (règle (b)). Lorsque la direction reste la même, la mise à jour est égale à la somme entre la mise à jour que AMSGrad aurait prise et la valeur $(1-\beta_1)m_{t-1}$ (règle (2)). Ainsi la borne du *regret* de AAMSGrad est :

$$
R_T \leq max(R_{T(a)}, R_{T(b)}) \tag{A.6}
$$

Où $R_{T(b)}$ est le *regret* quand on considère seulement la deuxième règle de mise à

jour d'AAMSGrad. Veuillez noter que la borne de $R_{T(b)}$ est similaire à celle de AMSGrad. $R_{T(a)}$ est le *regret* si nous considérons seulement la première règle. La preuve consistera à trouver la borne pour $R_{T(a)}$.

La première règle de mise à jour d'AAMSGrad peut être réécrite comme suit :

$$x_{t+1} \leq \Pi_{F,\sqrt{\hat{V}_t}}(x_t - U) \; where \; U = 2 \cdot \alpha_t \hat{V}_t^{-1/2} m_t \tag{A.7}$$

Si nous ignorons la valeur 2, la preuve sera la même que celle de la convergence d'AMSGrad. Ainsi, nous ne réécrivons pas la démonstration mais il est facile de voir qu'après avoir fait les mêmes étapes que Reddi et al, nous avons :

$$\begin{aligned} R_{T(a)} \leq & \frac{D_\infty^2}{2(1-\beta_1)} \sum_{i=1}^{d} (\frac{\hat{v}_{T,i}^{-1/2}}{\alpha_T} + \frac{\hat{v}_{2,i}^{-1/2}}{\alpha_2}) + \frac{D_\infty^2}{4(1-\beta_1)} \sum_{t=1}^{T} \sum_{i=1}^{d} \frac{\beta_{1t} \hat{v}_{t,i}^{1/2}}{\alpha_t} \\ & + \frac{2\alpha\sqrt{1+log(T)}}{(1-\beta_1)^2(1-\gamma)\sqrt{(1-\beta_2)}} \sum_{i=1}^{d} \left\| g_{1:T,i} \right\|_2 \end{aligned} \tag{A.8}$$

Parceque $R_{T(b)} \leq R_{T(a)}$, la borne du *regret* de AAMSGrad est :

$$R_T \leq max(R_{T(a)}, R_{T(b)}) = R_{T(a)} \tag{A.9}$$

□

Ce qui complète la peuve.

APPENDICE B

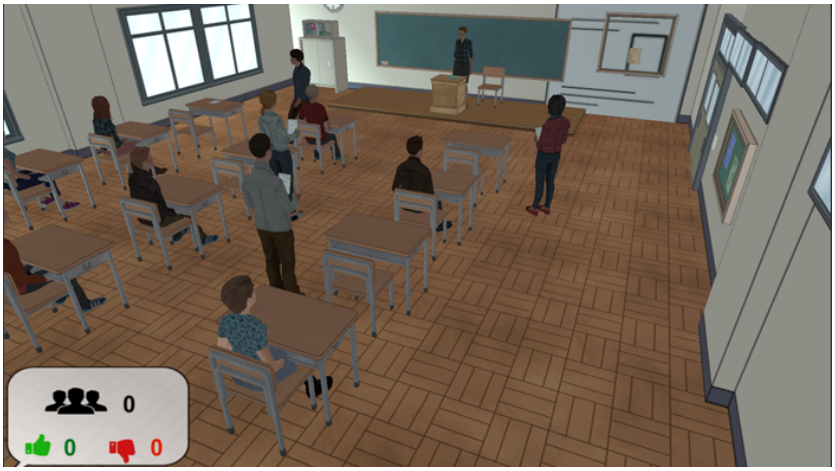
QUESTIONNAIRES : EXPÉRIMENTATION DU JEU LESDILEMMES

B.1 Questionnaire initial évaluant le style d'apprentissage et les habitudes de communication sur internet

Questionnaire initial - LesDilemmes

Nous aimerions savoir votre façon préférée d'apprendre et vos habitudes de communication sur internet.

*Obligatoire



1. **Surnom** *
   (ne l'oublie pas !)

2. **Numéro d'identification** *
   Donnez le numéro que nous vous avons fourni.

3. **Age** *
Une seule réponse possible.

- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- Plus de 19

4. **Sexe** *
Plusieurs réponses possibles.

- Fille
- Garcon
- Autre

Préférences pour l'apprentissage

Il n y a pas de bonnes ni de mauvaises réponses. Réponds le plus naturellement possible.

5. **Je me fixe des objectifs personnels qui dépassent ceux de mon enseignant** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord								Tout à fait d'accord

6. **J'ai du plaisir à apprendre** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord								Tout à fait d'accord

7. **Mon enseignant est la meilleure personne pour m'évaluer et me motiver** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord								Tout à fait d'accord

8. **J'aime apprendre sur des nouveaux sujets** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

9. **J'aime varier mes sources d'informations: télévision, magasine, Internet, amis** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

10. **Mon enseignant m'aide à bien comprendre le travail qui m'est demandé** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

11. **J'aime apprendre quand ça me sert à quelque chose** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

12. **J'évite de faire mes devoirs si je peux** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

13. **Je réussie mieux lorsque je m'entends bien avec mon enseignant** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

14. **Je préfère faire ce qui me plaît plutôt que de faire ce qui est demandé par mon enseignant** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

15. **Je n'ai pas besoin de raison, de tâche ou d'objectif pour apprendre** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

16. **J'apprends mieux lorsque je peux gérer moi-même les objectifs et les tâches du cours** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

17. **Je fais un plan avant d'accomplir une tâche** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

18. **Apprendre m'aide à relever des défis personnels** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

19. **J'évite, si je peux, une activité qui semble trop difficile** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

20. **Apprendre me fait sentir mieux préparé à affronter la vie** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

21. **J'aime suivre la progression de mes apprentissages, cela m'aide à m'améliorer** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

22. **Je me fixe des objectifs personnels qui dépassent ceux de mon cours** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

23. **J'ai du plaisir à découvrir ce qui m'intéresse** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

24. **J'ai du plaisir à découvrir ce qui m'intéresse** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

25. **Je me base sur mon enseignant pour l'évaluation de mes progrès** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

26. **J'évalue mes progrès personnels et je réfléchis à des moyens pour m'améliorer** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

27. **Mon enseignant est le mieux placé pour planifier mes apprentissages** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

28. **Je sais quoi faire si j'ai de la difficulté dans un cours** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

29. **Apprendre est désagréable et inconfortable pour moi** *
Une seule réponse possible.

	1	2	3	4	5	6	7	
Pas d'accord	◯	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

Vos habitudes de communication sur internet

30. **J'utilise des applications de partage comme Snapchat, Youtube, Instagram ou twitter.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

31. **Je vais régulièrement sur Facebook.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

32. **Mon nombre d'amis (ou suiveurs) sur Facebook, Instagram, Snapchat ou twitter n'est pas important pour moi.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

33. **J'aime mettre un « J'aime » sur les publications dans Facebook et Snapchat.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

34. **Je suis toujours d'accord avec ce que mes amis disent ou font.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

35. **Je préfère jouer à un jeu vidéo que d'aller jouer avec mes amis.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

36. **Je joue beaucoup aux jeux vidéo.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

37. **J'ai l'habitude de travailler sur un ordinateur.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

38. **Je joue souvent aux jeux vidéo qui permettent d'apprendre.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

B.2 Questionnaire final évaluant l'immersion, l'apprentissage et la satisfaction

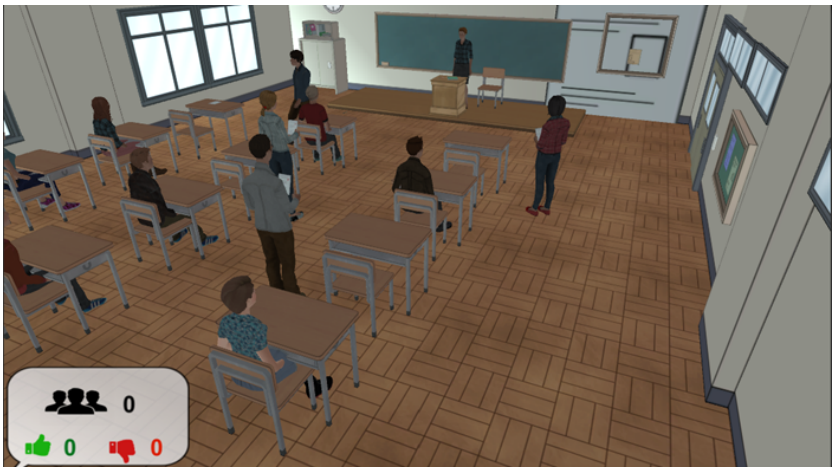
Jeu des dilemmes

Le jeu consiste en un ensemble de dilemmes sous forme d'un petit scénario dans lequel le participant est amené à prendre différentes décisions.

*Obligatoire

Questionnaire final

Il n y a pas de bonnes ni de mauvaises réponses. Réponds le plus naturellement possible.



1. **Surnom** *
   (ne l'oublie pas !)

2. **Numéro d'identification** *
   Donnez le numéro que nous vous avons fourni.

3. **Sexe** *
   Plusieurs réponses possibles.
   - [ ] Fille
   - [ ] Garcon
   - [ ] Autre

Immersion

4. **Je pensais à autre chose quand je jouais.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

5. **Parfois je voulais arrêter le jeu pour voir ce qui se passe autour de moi.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

6. **Le jeu m'a laissé indifférent.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

7. **J'ai déjà vécu les mêmes situations que celle du jeu dans la réalité.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

8. **J'étais parfois tellement impliqué que j'oubliais que j'étais dans un jeu.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

9. **J'ai aimé l'environnement et les personnages dans le jeu.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

10. **J'ai trouvé le jeu proche de la réalité.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

11. **Le jeu m'a fait réagir, j'étais confronté et embêté à répondre dans le jeu.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

Éfficacité du jeu

So-Moral 2017

12. **J'ai aimé jouer à ce jeu.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

13. **Je voudrais encore rejouer.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

14. **Je pense que mes amis aimeraient jouer à ce jeu.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

15. **J'ai aimé le fait que mon score dans le jeu était le nombre de « j'aime » que j'obtenais.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

16. **J'ai aimé le fait que mon score dans le jeu était le nombre d'amis que je gagnais.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

17. **J'ai trouvé interessant de pouvoir émettre mon opinion en parlant au micro.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

Évaluation du taux d'apprentissage

LesDilemmes 2019

18. **J'ai l'impression d'avoir appris des choses dans le jeu.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

19. **J'ai trouvé intéressant de lire et entendre les opinions des autres personnages du jeu.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

20. **J'ai trouvé intéressant de pouvoir dire que « j'aime » ou « je n'aime pas » les opinions des personnages du jeu.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

21. **Je vois les choses autrement qu'avant par rapport aux décisions à prendre.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

22. **Je comprends mieux l'impact de mes actes sur ce qui se passe.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

23. **Je pense avoir fait des progrès à la fin de la séance par rapport au début.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

24. **J'ai beaucoup réfléchi pendant le jeu.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

25. **Les commentaires dans le jeu m'ont permis d'apprendre quelque chose.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

26. **J'ai aimé que les amis dans le jeu réagissent quand j'évaluais ce qu'ils disaient.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

27. **Les réactions des autres joueurs (pouce vers le haut, vers le bas, émotions) m'ont influencé.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

28. **J'ai aimé le fait que la musique de fond change de temps en temps.** *
Une seule réponse possible.

	1	2	3	4	5	6	
Pas d'accord	◯	◯	◯	◯	◯	◯	Tout à fait d'accord

Questions ouvertes

Dis nous en quelques mots ce que tu as pensé de cet expérimentation en général.

29. **Qu’est ce que tu as aimé ?**

30. **Qu’est ce que tu n’as pas aimé ?**

APPENDICE C

MESSAGES D'ADAPTATION (DÉFINIS PAR LES EXPERTS) IMPLÉMENTÉS DANS LE JEU LESDILEMMES

C.1 Messages de félicitations :

- Bravo !
- C'est très bien.
- Félicitations !
- Excellent !
- Parfait.
- Bien, continue comme ça.
- Tu t'améliores.
- Super, tu t'en sors très bien
- Génial, continue comme ça !
- Tu apprends vite.
- Parfait, tu évolues bien.
- Félicitations, tu progresses.

C.2 Messages d'apprentissage :

Les messages d'Apprentissage devraient suggérer le développement à venir, ce que l'apprenant devrait faire pour accéder au niveau de connaissance suivant (niveau

de raisonnement supérieur).

Messages pour Niveau 1 (Orientation vers la punition et l'obéissance à l'autorité)–> Niveau 2 :

- *Dilemme 1* : As-tu pensé que cela peut être avantageux pour toi de respecter l'entente que tu avais avec ta mère ?
- *Dilemme 2* : Aimerais-tu qu'on te remettre ton portefeuille si tu l'échappais ?
- *Dilemme 3* : As-tu considéré que tu pourrais avoir un accident si tu traversais sur la lumière rouge ?
- *Dilemme 4* : Est-ce que tu apprécierais que ton ami rit de toi ?
- *Dilemme 5* : Est-ce possible que la professeur te ferait davantage confiance si tu ne trichais pas ?
- *Dilemme 6* : Est-ce possible que tu ne puisses plus retourner à cet endroit si tu volais ?
- *Dilemme 7* : Peut-être que tu te sentirais moins coupable si tu disais au propriétaire que tu as cassé son pare-brise par accident ...
- *Dilemme 8* : As-tu pensé que le chien pourrait te faire mal si tu l'agaçais ?
- *Dilemme 9* : As-tu considéré que de conserver ta note pourrait te permettre d'obtenir de l'aide de tes amis ?

Messages pour Niveau 2 (Orientation vers les échanges égocentriques) –> Niveau 3 :

- *Dilemme 1* : As-tu pensé que ta petite soeur pourrait avoir besoin qu'on s'occupe d'elle ?
- *Dilemme 2* : As-tu considéré que la personne qui a échappé son portefeuille apprécierait qu'on lui redonne ?
- *Dilemme 3* : As-tu pensé qu'un automobiliste pourrait avoir un accident à cause de ton comportement imprévisible ?

- *Dilemme 4* : Penses tu que ton ami se sentirait bien si tu ris de lui ?
- *Dilemme 5* : As-tu pensé que ton ami pourrait être puni par le professeur pour avoir triché ?
- *Dilemme 6* : Penses-tu que le propriétaire de l'épicerie aimerait qu'on lui vole des chocolats ?
- *Dilemme 7* : Est-ce que tu crois que le propriétaire aimerait savoir comment son pare-brise a été brisé ?
- *Dilemme 8* : As-tu pensé que le chien aurait mal si tu lui lançais des roches ?
- *Dilemme 9* : Penses tu que tes parents aimeraient savoir ta vraie note ?

Messages pour Niveau 3(Orientation vers les relations interpersonnelles) –> Niveau 4 :

- *Dilemme 1* : As-tu pensé que de respecter les ententes que l'on prend permet de conserver des relations harmonieuses ?
- *Dilemme 2* : As-tu considéré que le portefeuille appartient à la personne et que l'on devrait respecter sa propriété en lui redonnant ?
- *Dilemme 3* : As-tu réfléchi au fait que de respecter le code de la route permet d'assurer la sécurité de tous ?
- *Dilemme 4* : Crois-tu que tu respectes ton ami en riant de lui/elle ?
- *Dilemme 5* : As-tu tenu compte que les examens sont conçus pour évaluer les apprentissages réels des gens, et non ceux de leur voisin ?
- *Dilemme 6* : As-tu pensé que si tout le monde volait, les commerces ne pourraient pas fonctionner ?
- *Dilemme 7* : As-tu pensé que tu as endommagé la propriété de quelqu'un ?
- *Dilemme 8* : As-tu considéré que de lancer des roches serait de la cruauté animale ?
- *Dilemme 9* : As-tu pensé si tout le monde changeait sa note, les examens ne refléteraient plus le niveau d'apprentissage des gens ?

Niveau 4(Régulation de la société) et Niveau 5(Évaluation du contrat

social) : Messages d'apprentissage = Messages félicitations.

C.3 Messages de Généralisation

Les messages de généralisation sont des messages qui s'affichent lorsque qu'un joueur(se) atteint un niveau de raisonnement pour la première fois. Ils visent à rappeler au joueur les spécificitées du niveau qu'il vient d'atteindre.

- **Première fois le niveau 2 :** Yeah ! Tu as pensé aux avantages de la situation !
- **Première fois niveau 3 :** Bravo ! Tu as pensé à considérer les autres dans ta réponse au dilemme !
- **Première fois niveau 4 :** Tu as considéré le sens des règles dans la société pour prendre ta décision ! C'est super !
- **Première fois niveau 5 :** Super ! Tu considères plusieurs points de vue et tes valeurs pour prendre tes décisions !

APPENDICE D

RÈGLES D'ADAPTATION DU JEU LESDILEMMES

Niveau de raisonnement actuel = niveau calculé après justification R

Niveau de raisonnement cumulée = niveau de raisonnement moyen jusqu'au dilemme $t-1$ Niveau de raisonnement différence absolue = $\mid R_{t-1} - R_t \mid$

D.1 Règles définies à partir de la théorie

Déroulement du dilemme (Inner Loop) : Profil de l'apprenant et les réponses courantes

- Expressions non verbales des PNJs en fonction de leur évaluation par le joueur-apprenant :
  - **Si** d'accord avec un PNJ de niveau supérieur $R_x >= R$ **Alors** émotion positive (Content, souriant , pouce vers le haut).
  - **Si** d'accord avec un PNJ de niveau $R_x < R$ **Alors** ne rien faire (Visage neutre).
  - **Si** d'accord avec un PNJ de niveau $R_x < R$ **Alors** ne rien faire (Visage neutre).
  - Si pas d'accord avec un PNJ de niveau $R_x >= R$ **Alors** émotion négative (Faché, pousse vers le bas).

- Messages félicitations : Féliciter le joueur lorsqu'il évolue bien en général :
  - **Si** d'accord avec un PNJ de niveau supérieur $R_x >= R$ **Alors** messages d'encouragements ou félicitations sur le niveau actuel (permet de pour renforcer et faciliter le transfert).
  - **Si** le niveau de raisonnement est en progrès **Alors** messages de félicitations sur le niveau actuel.
  - **Si** le niveau de raisonnement n'est pas en progrès **Alors** passer au point (3) ci-dessous
- Messages d'apprentissage en fonction du niveau de raisonnement (après justification du joueur) :
  - **Si** différence absolue positif **Alors** message félicitations + message d'apprentissage qui dépends de son niveau s'il n'a pas progressé et suggère attitude pour niveau suivant.
  - **Si** non **Alors** message d'encouragements + message d'apprentissage qui dépends de son niveau de raisonnement.
  - **Si** le joueur est de niveau : **1** (Peur de l'autorité) **Alors** message qui l'aidera à penser comme quelqu'un de niveau 2 ou 3. (le choix du message est automatique dans le jeu). **2** (égoiste) **Alors** message qui l'aidera à penser comme quelqu'un de niveau 3 ou 4. **3** (relations interpersonnelles) **Alors** message qui l'aidera à penser comme quelqu'un de niveau 4 ou 5.
- Adaptation en fonction des émotions actuelles ou prédits du joueur :
  - **Si** émotions positives (Happy, Neutre) **Alors** musique = douce, joviale
  - **Si** émotions négatives (Sad, Angry, Disgusted) **Alors** musique = Joviale, dansante, drôle

Construction du dilemme (OuterLoop) :

- **Si** le joueur visite uniquement les joueurs de niveau faible **Alors** (l'inciter à

aller voir ce que pensent les autres de niveau supérieur en forçant d'aller voir tout le monde ou en enlevant les PNJs non nécessaires lors des prochains dilemmes) nombre de PNJ à visiter = 5.

- Changer le mode de calcul du score en fonction du style d'apprentissage :
  - **Si** style d'apprentissage = Performant **Alors** nombre amis = niveau de raisonnement * 2.
  - **Si** style d'apprentissage = Conformiste **Alors** nombre amis = niveau de raisonnement.
  - **Si** style d'apprentissage = Résistant ou Intentionnel **Alors** nombre amis = niveau de raisonnement.

D.2 Règles définies à partir de la théorie

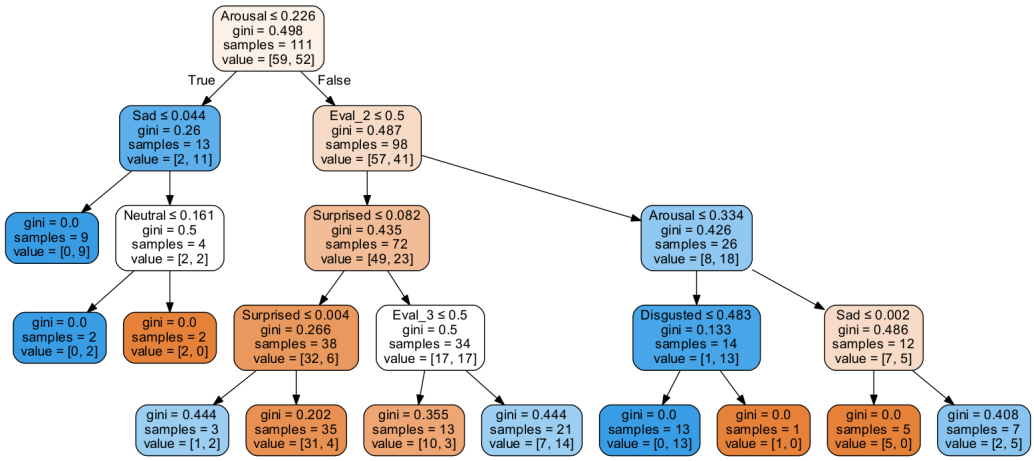


Figure D.1 Arbre de décision appris à partir des données de la première expérimentation sur le jeu non adaptatif.

Règles extraits de l'arbre de décision (figure D.1) et du RN (figure D.2) générés à partir des données récoltées pendant la première expérimentation sur le jeu non adaptatif :

- **Si** Arousal < 0.228 et Sad < 0.02 **Alors** Musique douce.

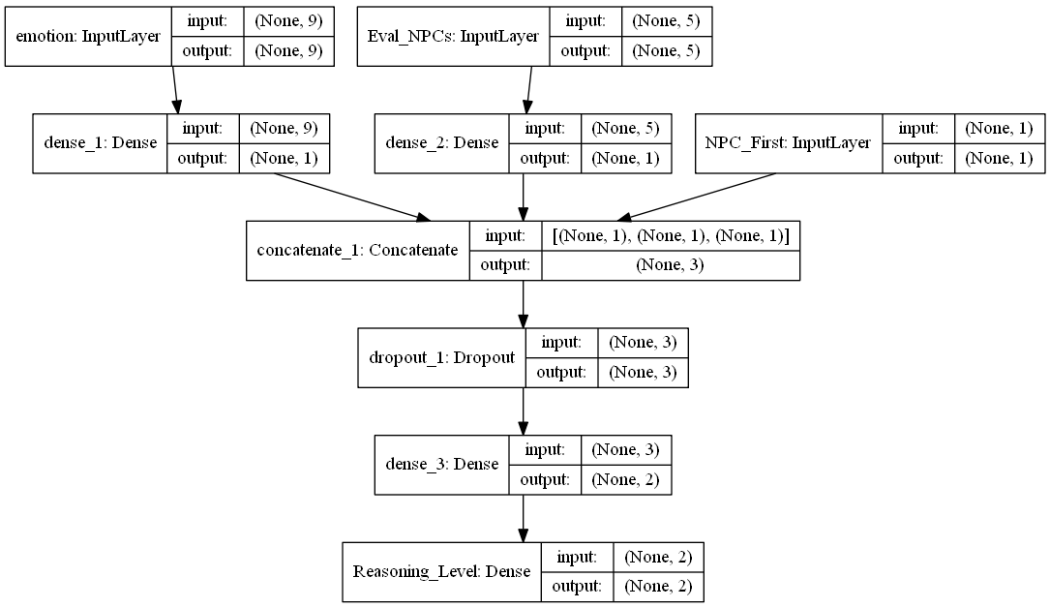


Figure D.2 Le réseau de neurones multimodale proposé pour l'extraction des règles d'adaptation.

- **Si** le joueur n'a pas visité les PNJs de niveau 2 ou 5, **Alors** l'obliger à le faire dans les prochains dilemmes.

APPENDICE E

EXTRACTION DES RÈGLES D'ADAPTATION PAR RÉSEAU DE NEURONES : LES POIDS APPRIS

Tableau E.1 Les poids du réseau de neurones multimodal entraîné (W poids reliant les neurones de l'avant dernière couche et la dernière couche) appris à partir de 3 entraînements différents. $A_{i,j}$ est calculé après 20 runs.

Runs	Run 1		Run 2		Run 3		$A_{i,1}$	$A_{i,2}$
Output	[1,0]	[0,1]	[1,0]	[0,1]	[1,0]	[0,1]	[1,0]	[0,1]
`Emotions`	0.007	0.812	0	0.751	0.925	0.739	0.475	0.689
`Eval NPCs`	0.066	0	0.3231	0.040	0.043	0.378	0.367	0.276
`First NPC visited`	0.067	0.607	0.742	0.099	0.116	0.071	0.158	0.035

E.1 Comparaison des émotions entre la version non adaptative et la version adaptative du jeu LesDilemmes

Tableau E.2 Les poids du réseau de neurones multimodal entraîné (W poids du neurone qui représentent les émotiions) appris à partir de 3 entraînements différents. A_i est calculé après 20 runs.

Neuron (Emotions)	Run 1	Run 2	Run 3	A_i
`Neutral`	0.080	0.0.093	0	0.001
`Happy`	0	0.4748	0	0.120
`Sad`	0.672	0.013	0.333	0.165
`Angry`	0.069	0.702	0	0.090
`Surprised`	0	0.034	0.583	0.155
`Scared`	0.138	0	0.041	0.001
`Disgusted`	0	0.016	0.001	0.106
`Valence`	0.346	0.006	0.045	0.128
`Arousal`	0.039	0.404	0.578	0.234

Tableau E.3 Les poids du réseau de neurones multimodal entraîné (W poids du neurones représentant l'évaluation des 5 PNJs) appris à partir de 3 entraînements différents. A_i est calculé après 20 runs.

Neuron (Eval NPCs)	Run 1	Run 2	Run 3	A_i
`Eval 1`	0.939	0.186	0	0.198
`Eval 2`	0.0.078	0.755	0	0.215
`Eval 3`	0.052	0	0.938	0,167
`Eval 4`	0.028	0.588	0	0,147
`Eval 5`	0.920	0.369	0	0.273

Statistiques de groupe

	Adaptive	N	Moyenne	Ecart type	Moyenne erreur standard
Neutral	A	203	.525031595	.150881620	.010589814
	NA	144	.192085526	.105721967	.008810164
Happy	A	203	.078216982	.094744399	.006649753
	NA	144	.089506325	.098287123	.008190594
Sad	A	203	.197926328	.174387221	.012239583
	NA	144	.040159058	.123057789	.010254816
Angry	A	203	.131614837	.137321689	.009638093
	NA	144	.459101220	.291906340	.024325528
Surprised	A	203	.146550201	.117056987	.008215790
	NA	144	.104515495	.140334336	.011694528
Scared	A	203	.084387812	.071013903	.004984199
	NA	144	.010575388	.049373191	.004114433
Disgusted	A	203	.055441721	.054976442	.003858590
	NA	144	.159269417	.222573397	.018547783
Valence	A	203	-.25367139	.216995381	.015230090
	NA	144	-.49613987	.314730717	.026227560
Arousal	A	203	.332676967	.064513975	.004527993
	NA	144	.283627665	.079556000	.006629667

Figure E.1 Statistiques de groupe : Version adaptative (A) versus version non adaptative (NA) du jeu LesDilemmes.

Test des échantillons indépendants

		Test de Levene sur l'égalité des variances		Test t pour égalité des moyennes						
									Intervalle de confiance de la différence à 95 %	
		F	Sig.	t	ddl	Sig. (bilatéral)	Différence moyenne	Différence erreur standard	Inférieur	Supérieur
Neutral	Hypothèse de variances égales	10.538	.001	22.801	345	.000	.332946069	.014602043	.304225837	.361666301
	Hypothèse de variances inégales			24.170	344.958	.000	.332946069	.013775454	.305851613	.360040525
Happy	Hypothèse de variances égales	1.789	.182	-1.077	345	.282	-.01128934	.010484311	-.03191056	.009331871
	Hypothèse de variances inégales			-1.070	301.050	.285	-.01128934	.010550120	-.03205066	.009471977
Sad	Hypothèse de variances égales	33.051	.000	9.331	345	.000	.157767270	.016907788	.124511953	.191022588
	Hypothèse de variances inégales			9.880	344.994	.000	.157767270	.015967738	.126360901	.189173639
Angry	Hypothèse de variances égales	140.220	.000	-13.960	345	.000	-.32748638	.023458798	-.37362665	-.28134612
	Hypothèse de variances inégales			-12.516	188.139	.000	-.32748638	.026165324	-.37910149	-.27587127
Surprised	Hypothèse de variances égales	.101	.751	3.033	345	.003	.042034706	.013861223	.014771566	.069297845
	Hypothèse de variances inégales			2.941	272.072	.004	.042034706	.014291997	.013897743	.070171668
Scared	Hypothèse de variances égales	60.537	.000	10.762	345	.000	.073812424	.006858887	.060321927	.087302922
	Hypothèse de variances inégales			11.421	344.879	.000	.073812424	.006463033	.061100502	.086524346
Disgusted	Hypothèse de variances égales	126.426	.000	-6.381	345	.000	-.10382770	.016271168	-.13583087	-.07182452
	Hypothèse de variances inégales			-5.481	155.439	.000	-.10382770	.018944893	-.14125036	-.06640503
Valence	Hypothèse de variances égales	38.238	.000	8.495	345	.000	.242468473	.028542017	.186330209	.298606736
	Hypothèse de variances inégales			7.995	236.650	.000	.242468473	.030328873	.182719412	.302217533
Arousal	Hypothèse de variances égales	.858	.355	6.329	345	.000	.049049301	.007750392	.033805334	.064293268
	Hypothèse de variances inégales			6.109	266.479	.000	.049049301	.008028400	.033242135	.064856467

Figure E.2 Test des échantillons indépendants : Comparaison des moyennes entre les émotions sur la version adaptative (A) et non adaptative (NA) du jeu LesDilemmes.

APPENDICE F

QUELQUES CAPTURES D'ÉCRAN DU JEU LESDILEMMES

Figure F.1 Une capture du premier dilemme du jeu. Affichage du gain de "likes" et "dislikes" après l'évaluation du justificatif du joueur. Ici le joueur a donné un argument de niveau de raisonnement 1.

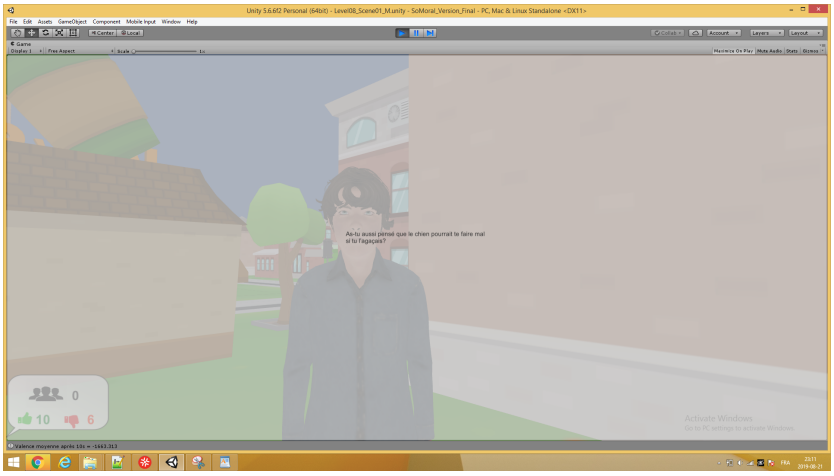


Figure F.2 Affichage d'un message d'apprentissage dans le jeu.

```
using System.Collections;
using System.Collections.Generic;
using System;
using System.Linq;
using UnityEngine;
using UnityEngine.SceneManagement;
using System.IO;

public class PlayerModel : MonoBehaviour {

    private static bool created = false; // Useful for the DontDestroyOnLoad in the Awake() function : allow this object to exist until the end of the game

    private static string path_data_export = @"PlayerModelDatas\";

    private static string player_gender = "F";

    public static float limiteEmotions = 500;
    private static float[] emotions = new float[9] { 0.0f, 0.0f, 0.0f, 0.0f, 0.0f, 0.0f, 0.0f, 0.0f, 0.0f }; // (Neutral Happy       Sad       Angry Surprised       Scared Disgusted       Valence Arousal)
    private static List<float[]> emotionsList = new List<float[]>();
    private static int currentDilemmeEmotionsPosition = 0;

    private static int decision;
    private static float decision_time;
    private static List<int> decisions = new List<int>();  // liste de ses décisions à tous les dilemmes(S'accumule). De la forme [1,0,1,1,0...], 1 pour en accord et 0 pour le contraire
    private static List<float> decisions_time = new List<float>();


    private static float reasoning_level = 0;
    private static List<float> reasoning_levels = new List<float>();

    private static List<int> npc_order = new List<int>();
    private static List<int> npc_decisions = new List<int>();
    private static List<float> npc_decisions_time = new List<float>();

    private static int pnj_number_to_visit = 3;

    public static string learning_style;

    private static string music;
    private static List<string> musicList = new List<string>();

    private static int totalDilemmeNumber = 9;
    private static string dilemme = "";
    private static List<String> remainingDilemme = new List<String>();
    private static string next_dilemme = "";

    private static int nbAmis {get; set; }
    private static int nbreLikesCum { get; set; }
    private static int nbreDislikesCum { get; set; }
```

Figure F.3 Extrait d'une classe codée en c#, représentant le modèle de l'usager tel que implémenté dans le jeu LesDilemmes.

APPENDICE G

MUSE-LOGIQUE

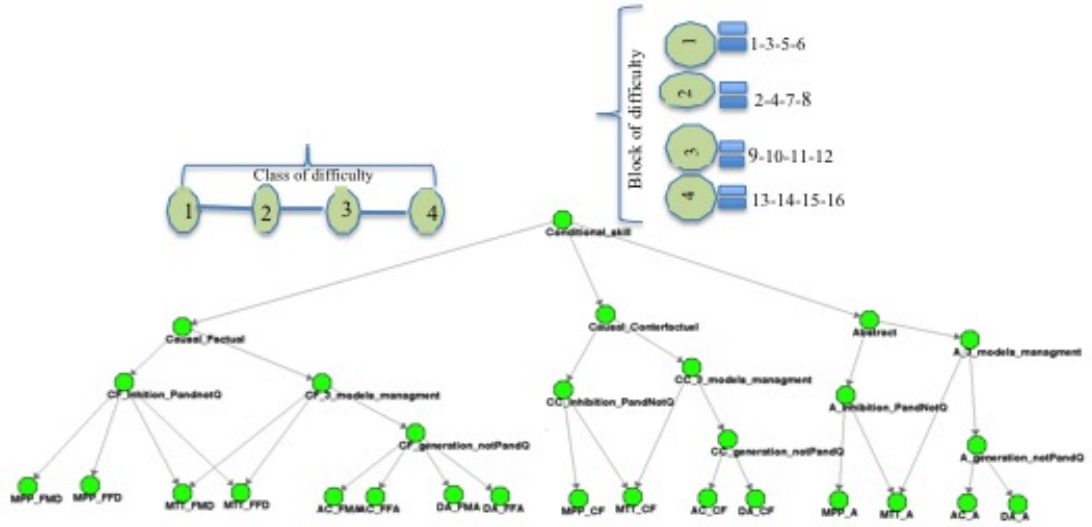


Figure G.1 Représentation des problèmes de raisonnement logique dans le réseau bayésien construit par les experts du domaine.

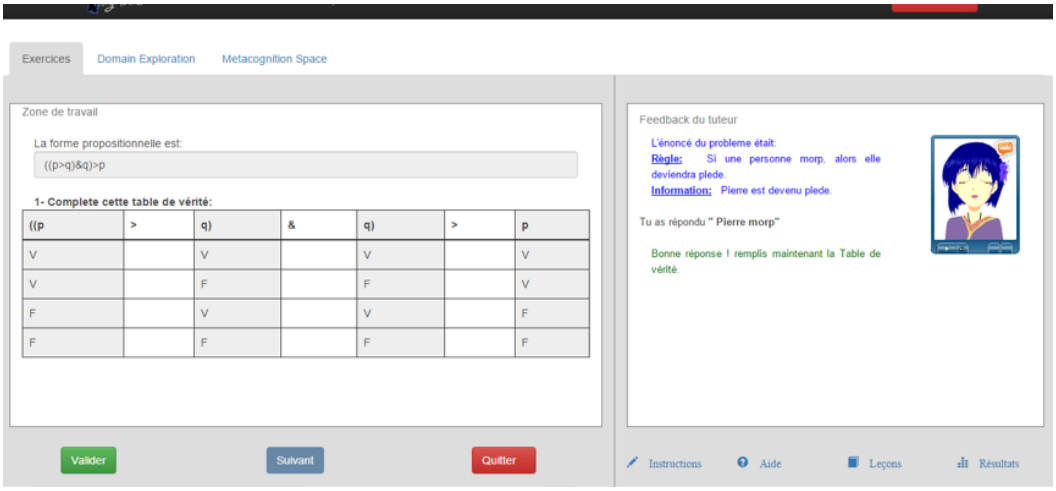


Figure G.2 Interface apprenant – Construction Table de vérité après mauvaise réponse

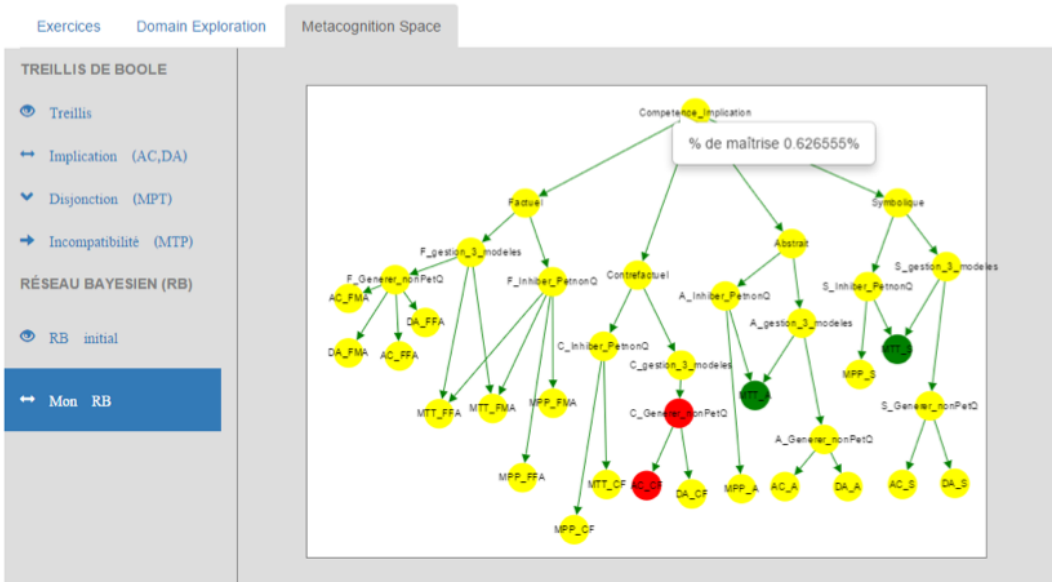


Figure G.3 Interface apprenant – Visualisation du Réseau bayésien

RÉFÉRENCES

Abdel-Hamid, O., Mohamed, A.-r., Jiang, H., Deng, L., Penn, G. et Yu, D. (2014). Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, *22*(10), 1533–1545.

Abyaa, A., Idrissi, M. K. et Bennani, S. (2018). Predicting the learner's personality from educational data using supervised learning. Dans *Proceedings of the 12th International Conference on Intelligent Systems : Theories and Applications*, p. 19. ACM.

Adler, J. E. (2008). Surprise. *Educational Theory*, *58*(2), 149–173.

Ahmed, A., Hong, L. et Smola, A. J. (2013). Hierarchical geographical modeling of user locations from social media posts. Dans *Proceedings of the 22nd international conference on World Wide Web*, 25–36. ACM.

Al-Abri, A., Muscat, O., AlKhanjari, Z., Jamoussi, Y. et Kraiem, N. (2019). User modeling for personalized e-learning based on social collaboration interaction. Dans *4th FREE & OPEN SOURCE SOFTWARE CONFERENCE (FOSSC'2019-OMAN)*.

Al-Nakhal, M. A. et Naser, S. S. A. (2017). Adaptive intelligent tutoring system for learning computer theory.

Al Rekhawi, H. A. et Abu Naser, S. S. (2018). An intelligent tutoring system for learning android applications ui development. *International Journal of Engineering and Information Systems (IJEAIS)*, *2*(1), 1–14.

Albert, D. et Lukas, J. (1999). *Knowledge spaces : Theories, empirical research, and applications*. Psychology Press.

Alexander, J. T., Sear, J. et Oikonomou, A. (2013). An investigation of the effects of game difficulty on player enjoyment. *Entertainment computing*, *4*(1), 53–62.

Alitalo, T. (2016). Using facereader to recognize emotions during self-assessment relating to dyslexia.

Andersen, E., Gulwani, S. et Popovic, Z. (2013). A trace-based framework for analyzing and synthesizing educational progressions. Dans *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 773–782. ACM.

Anderson, J. R. (1996). Act : A simple theory of complex cognition. *American psychologist*, *51*(4), 355.

Anderson, J. R., Corbett, A. T., Fincham, J. M., Hoffman, D. et Pelletier, R. (1992). General principles for an intelligent tutoring architecture. *Cognitive approaches to automated instruction*, 81–106.

Anderson, J. R., Matessa, M. et Lebiere, C. (1997). Act-r : A theory of higher level cognition and its relation to visual attention. *Human-Computer Interaction*, *12*(4), 439–462.

Arnold, S., Fujima, J., Karsten, A. et Simeit, H. (2013). Adaptive behavior with user modeling and storyboarding in serious games. Dans *2013 International Conference on Signal-Image Technology & Internet-Based Systems*, 345–350. IEEE.

Atkinson, R. C. et Shiffrin, R. M. (1968). Human memory : A proposed system and its control processes1. In *Psychology of learning and motivation*, volume 2 89–195. Elsevier.

Bae, B.-C. et Young, R. M. (2009). Suspense ? surprise ! or how to generate stories with surprise endings by exploiting the disparity of knowledge between a story's reader and its characters. Dans *Joint International Conference on Interactive Digital Storytelling*, 304–307. Springer.

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

Bakkes, S. C., Spronck, P. H. et van Lankveld, G. (2012). Player behavioural modelling for video games. *Entertainment Computing*, *3*(3), 71–79.

Barrett, L. F. (1998). Discrete emotions or dimensions ? the role of valence focus and arousal focus. *Cognition & Emotion*, *12*(4), 579–599.

Bartle, R. (1996). Hearts, clubs, diamonds, spades : Players who suit muds. *Journal of MUD research*, *1*(1), 19.

Beauchamp, M., Dooley, J. J. et Anderson, V. (2013). A preliminary investigation of moral reasoning and empathy after traumatic brain injury in adolescents. *Brain injury*, *27*(7-8), 896–902.

Beck, J. E. et Chang, K.-m. (2007). Identifiability : A fundamental problem of student modeling. Dans *International Conference on User Modeling*, 137–146. Springer.

Benyon, D. et Murray, D. (1993). Applying user modeling to human-computer interaction design. *Artificial Intelligence Review*, *7*(3-4), 199–225.

Bernstein, J., Wang, Y.-X., Azizzadenesheli, K. et Anandkumar, A. (2018). signsgd : compressed optimisation for non-convex problems. *arXiv preprint arXiv :1802.04434*.

Billsus, D. et Pazzani, M. J. (2000). User modeling for adaptive news access. *User modeling and user-adapted interaction*, *10*(2-3), 147–180.

Birk, M. V., Toker, D., Mandryk, R. L. et Conati, C. (2015). Modeling motivation in a social network game using player-centric traits and personality traits. Dans *International Conference on User Modeling, Adaptation, and Personalization*, 18–30. Springer.

Blom, P. M., Bakkes, S., Tan, C. T., Whiteson, S., Roijers, D., Valenti, R. et Gevers, T. (2014). Towards personalised gaming via facial expression recognition. Dans *Tenth Artificial Intelligence and Interactive Digital Entertainment Conference*.

Bologna, G. et Hayashi, Y. (2018). A comparison study on rule extraction from neural network ensembles, boosted shallow trees, and svms. *Applied Computational Intelligence and Soft Computing*, *2018*.

Bontchev, B. (2016). Holistic player modelling for controlling adaptation in video games. *e-Society 2016*, p. 11.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010* 177–186. Springer.

Brisson, A., Pereira, G., Prada, R., Paiva, A., Louchart, S., Suttie, N., Lim, T., Lopes, R. A., Bidarra, R., Bellotti, F. *et al.* (2012). Artificial intelligence and personalization opportunities for serious games. Dans *Eighth Artificial Intelligence and Interactive Digital Entertainment Conference*.

Brusilovsky, P. et Millán, E. (2007). User models for adaptive hypermedia and adaptive educational systems. In *The adaptive web* 3–53. Springer.

Butler, E., Andersen, E., Smith, A. M., Gulwani, S. et Popović, Z. (2015). Automatic game progression design through analysis of solution features. Dans *Proceedings of the 33rd annual ACM conference on human factors in computing systems*, 2407–2416. ACM.

Cai, X., Hu, S. et Lin, X. (2012). Feature extraction using restricted boltzmann machine for stock price prediction. Dans *2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE)*, volume 3, 80–83. IEEE.

Carr, B. et Goldstein, I. P. (1977). *Overlays : A theory of modelling for computer aided instruction*. Rapport technique, MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB.

Charles, D. et Black, M. (2004). Dynamic player modeling : A framework for player-centered digital games. Dans *Proc. of the International Conference on Computer Games : Artificial Intelligence, Design and Education*, 29–35.

Charles, D., Kerr, A., McNeill, M., McAlister, M., Black, M., Kcklich, J., Moore, A. et Stringer, K. (2005). Player-centred game design : Player modelling and adaptive digital games. Dans *Proceedings of the digital games research conference*, volume 285, p. 00100.

Cheng, H.-T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M. *et al.* (2016). Wide & deep learning for recommender systems. Dans *Proceedings of the 1st workshop on deep learning for recommender systems*, 7–10. ACM.

Chiasson, V., Vera-Estay, E., Lalonde, G., Dooley, J. et Beauchamp, M. (2017). Assessing social cognition : age-related changes in moral reasoning in childhood and adolescence. *The Clinical Neuropsychologist*, *31*(3), 515–530.

Chorowski, J. et Zurada, J. M. (2015). Learning understandable neural networks with nonnegative weight constraints. *IEEE transactions on neural networks and learning systems*, *26*(1), 62–69.

Chorowski, J. K., Bahdanau, D., Serdyuk, D., Cho, K. et Bengio, Y. (2015). Attention-based models for speech recognition. Dans *Advances in neural information processing systems*, 577–585.

Conati, C., Gertner, A. et Vanlehn, K. (2002). Using bayesian networks to manage uncertainty in student modeling. *User modeling and user-adapted interaction*, *12*(4), 371–417.

Conati, C., Gertner, A. S., VanLehn, K. et Druzdzel, M. J. (1997). On-line student modeling for coached problem solving using bayesian networks. Dans *User Modeling*, 231–242. Springer.

Conati, C., Lallé, S., Rahman, M. A. et Toker, D. (2017). Further results on

predicting cognitive abilities for adaptive visualizations. Dans *IJCAI*, 1568–1574.

Conati, C. et Maclaren, H. (2009). Empirically building and evaluating a probabilistic model of user affect. *User Modeling and User-Adapted Interaction*, *19*(3), 267–303.

Cooke, N. J. (1999). Knowledge elicitation. *Handbook of applied cognition*, 479–509.

Corbett, A. T. et Anderson, J. R. (1994). Knowledge tracing : Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, *4*(4), 253–278.

Cordón, O., Herrera, F. et Villar, P. (2001). Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base. *IEEE Transactions on fuzzy systems*, *9*(4), 667–674.

Coro, G., Pagano, P. et Ellenbroek, A. (2013). Combining simulated expert knowledge with neural networks to produce ecological niche models for latimeria chalumnae. *Ecological modelling*, *268*, 55–63.

Damashek, M. (1995). Gauging similarity with n-grams : Language-independent categorization of text. *Science*, *267*(5199), 843–848.

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., Senior, A., Tucker, P., Yang, K., Le, Q. V. *et al.* (2012). Large scale distributed deep networks. Dans *Advances in neural information processing systems*, 1223–1231.

Demuth, H. B., Beale, M. H., De Jess, O. et Hagan, M. T. (2014). *Neural network design*. Martin Hagan.

Devillers, L. et Vidrascu, L. (2006). Real-life emotions detection with lexical and paralinguistic cues on human-human call center dialogs. Dans *Ninth International Conference on Spoken Language Processing*.

Dietterich, T. G. (2000). Ensemble methods in machine learning. Dans *International workshop on multiple classifier systems*, 1–15. Springer.

Digman, J. M. (1990). Personality structure : Emergence of the five-factor model. *Annual review of psychology*, *41*(1), 417–440.

Dooley, J. J., Beauchamp, M. et Anderson, V. A. (2010). The measurement of sociomoral reasoning in adolescents with traumatic brain injury : A pilot investigation. *Brain Impairment*, *11*(2), 152–161.

Dozat, T. (2016). Incorporating nesterov momentum into adam.

Drachen, A., Green, J., Gray, C., Harik, E., Lu, P., Sifa, R. et Klabjan, D. (2016). Guns and guardians : Comparative cluster analysis and behavioral profiling in destiny. Dans *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, 1–8. IEEE.

Drachen, A., Sifa, R., Bauckhage, C. et Thurau, C. (2012). Guns, swords and data : Clustering of player behavior in computer games in the wild. Dans *2012 IEEE conference on Computational Intelligence and Games (CIG)*, 163–170. IEEE.

Duchi, J., Hazan, E. et Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, *12*(Jul), 2121–2159.

Duchowski, A. T. (2007). Eye tracking methodology. *Theory and practice*, *328*(614), 2–3.

D'mello, S. K., Craig, S. D., Witherspoon, A., Mcdaniel, B. et Graesser, A. (2008). Automatic detection of learner's affect from conversational cues. *User modeling and user-adapted interaction*, *18*(1-2), 45–80.

Eagle, M. (2009). Level up : a frame work for the design and evaluation of educational games. Dans *Proceedings of the 4th International Conference on Foundations of Digital Games*, 339–341. ACM.

Eberhart, R. et Kennedy, J. (1995). A new optimizer using particle swarm theory. Dans *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*, 39–43. IEEE.

Ekman, P. (1999). Basic emotions. *Handbook of cognition and emotion*, *98*(45-60), 16.

Ekstrand, M. D., Riedl, J. T., Konstan, J. A. *et al.* (2011). Collaborative filtering recommender systems. *Foundations and Trends® in Human–Computer Interaction*, *4*(2), 81–173.

Elkahky, A. M., Song, Y. et He, X. (2015). A multi-view deep learning approach for cross domain user modeling in recommendation systems. Dans *Proceedings of the 24th International Conference on World Wide Web*, 278–288. International World Wide Web Conferences Steering Committee.

Evans, J. S. B., Newstead, S. E., Byrne, R. M. *et al.* (1993). *Human reasoning : The psychology of deduction*. Psychology Press.

Ezhilarasi, R. et Minu, R. (2012). Automatic emotion recognition and classification. *Procedia Engineering*, *38*, 21–26.

Farooq, S. et Kim, K. (2016). Game player modeling, encyclopedia of computer graphics and games.

Felder, R. M., Silverman, L. K. *et al.* (1988). Learning and teaching styles in engineering education. *Engineering education*, *78*(7), 674–681.

Feldman, L. A. (1995). Valence focus and arousal focus : Individual differences in the structure of affective experience. *Journal of personality and social psychology*, *69*(1), 153.

Fink, J. et Kobsa, A. (2002). User modeling for personalized city tours. *Artificial intelligence review*, *18*(1), 33–74.

Fischer, G. (2001). User modeling in human–computer interaction. *User modeling and user-adapted interaction*, *11*(1-2), 65–86.

Fisher, J. L., Hinds, S. C. et D'Amato, D. P. (1990). A rule-based system for document image segmentation. Dans *[1990] Proceedings. 10th International Conference on Pattern Recognition*, volume 1, 567–572. IEEE.

Ford, D. N. et Sterman, J. D. (1998). Expert knowledge elicitation to improve formal and mental models. *System Dynamics Review : The Journal of the System Dynamics Society*, *14*(4), 309–340.

Foster, M. I. et Keane, M. T. (2019). The role of surprise in learning : Different surprising outcomes affect memorability differentially. *Topics in cognitive science*, *11*(1), 75–87.

Frias-Martinez, E., Cebrian, M., Pascual, J. M. et Oliver, N. (2009). Explicit vs. implicit tagging for user modeling. Dans *PMPC@ UMAP*.

Fridgen, M., Schackmann, J. et Volkert, S. (2000). Preference based customer models for electronic banking.

Friedman, N., Nachman, I. et Peér, D. (1999). Learning bayesian network structure from massive datasets : the «sparse candidate «algorithm. Dans *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, 206–215. Morgan Kaufmann Publishers Inc.

Frome, J. (2007). Eight ways videogames generate emotion. Dans *DiGRA conference*.

Froschauer, J., Seidel, I., Gärtner, M., Berger, H. et Merkl, D. (2010). Design and evaluation of a serious game for immersive cultural training. Dans *2010*

16th international conference on virtual systems and multimedia, 253–260. IEEE.

Fu, Y., Hospedales, T. M., Xiang, T. et Gong, S. (2013). Learning multimodal latent attributes. *IEEE transactions on pattern analysis and machine intelligence*, *36*(2), 303–316.

Garcia, I., Sebastia, L. et Onaindia, E. (2011). On the design of individual and group recommender systems for tourism. *Expert systems with applications*, *38*(6), 7683–7692.

Ghali, R., Abdessalem, H. B., Frasson, C. et Nkambou, R. (2018). Identifying brain characteristics of bright students. *Journal of Intelligent Learning Systems and Applications*, *10*(03), 93.

Giardina, M. et Laurier, M. (1999). Modélisation de l'apprenant et interactivité. *Revue des sciences de l'éducation*, *25*(1), 35–59.

Göbel, S. et Mehm, F. (2013). Personalized, adaptive digital educational games using narrative game-based learning objects. In *Serious Games and Virtual Worlds in Education, Professional Development, and Healthcare* 74–84. IGI Global.

Goodfellow, I., Bengio, Y., Courville, A. et Bengio, Y. (2016). *Deep learning*, volume 1. MIT Press.

Gow, J., Baumgarten, R., Cairns, P., Colton, S. et Miller, P. (2012). Unsupervised modeling of player style with lda. *IEEE Transactions on Computational Intelligence and AI in Games*, *4*(3), 152–166.

Graves, A. (2013). Generating sequences with recurrent neural networks. *arXiv preprint arXiv :1308.0850*.

Graves, A., Mohamed, A.-r. et Hinton, G. (2013). Speech recognition with deep recurrent neural networks. Dans *2013 IEEE international conference on acoustics, speech and signal processing*, 6645–6649. IEEE.

Greenberg, R. (2010). *Kant's theory of a priori knowledge*. Penn State Press.

Gunes, H. et Piccardi, M. (2007). Bi-modal emotion recognition from expressive face and body gestures. *Journal of Network and Computer Applications*, *30*(4), 1334–1345.

Ha, E. Y., Rowe, J. P., Mott, B. W. et Lester, J. C. (2011). Goal recognition with markov logic networks for player-adaptive games. Dans *Seventh Artificial Intelligence and Interactive Digital Entertainment Conference*.

Halpern, D., Tubridy, S., Wang, H. Y., Gasser, C., Popp, P. O., Davachi, L. et Gureckis, T. M. (2018). Knowledge tracing using the brain. *International Educational Data Mining Society*.

Hamari, J. et Tuunanen, J. (2014). Player types : A meta-synthesis.

He, H. et Garcia, E. A. (2008). Learning from imbalanced data. *IEEE Transactions on Knowledge & Data Engineering*, (9), 1263–1284.

He, X., Liao, L., Zhang, H., Nie, L., Hu, X. et Chua, T.-S. (2017). Neural collaborative filtering. Dans *Proceedings of the 26th international conference on world wide web*, 173–182. International World Wide Web Conferences Steering Committee.

Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J. et Scholkopf, B. (1998). Support vector machines. *IEEE Intelligent Systems and their applications*, *13*(4), 18–28.

Hendrikx, M., Meijer, S., Van Der Velden, J. et Iosup, A. (2013). Procedural content generation for games : A survey. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, *9*(1), 1.

Henze, N. et Herder, E. (2011). User Modeling and Personalization 3 : User Modeling - Techniques. `https://www.eelcoherder.com/images/teaching/usermodeling/03_user_modeling_techniques.pdf`. Dernière visite 2019-07-14.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural computation*, *14*(8), 1771–1800.

Hinton, G. E. (2009). Deep belief networks. *Scholarpedia*, *4*(5), 5947.

Hinton, G. E. et Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *science*, *313*(5786), 504–507.

Hochreiter, S. et Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Horvitz, E., Breese, J., Heckerman, D., Hovel, D. et Rommelse, K. (1998). The lumiere project : Bayesian user modeling for inferring the goals and needs of software users. Dans *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, 256–265. Morgan Kaufmann Publishers Inc.

Isola, P., Zhu, J.-Y., Zhou, T. et Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. *arXiv preprint*.

Jaques, N., Conati, C., Harley, J. M. et Azevedo, R. (2014). Predicting affect from gaze data during interaction with an intelligent tutoring system. Dans *International conference on intelligent tutoring systems*, 29–38. Springer.

Jatupaiboon, N., Pan-ngum, S. et Israsena, P. (2013). Real-time eeg-based happiness detection system. *The Scientific World Journal*, *2013*.

Jensen, F. V. *et al.* (1996). *An introduction to Bayesian networks*, volume 210. UCL press London.

Jerčić, P., Astor, P. J., Adam, M., Hilborn, O., Schaff, K., Lindley, C., Sennersten, C. et Eriksson, J. (2012). A serious game using physiological interfaces for emotion regulation training in the context of financial decision-making. Dans *20th European Conference on Information Systems (ECIS 2012)*, 1–14. AIS Electronic Library (AISeL).

Jraidi, I. (2014). Modélisation des émotions de l'apprenant et interventions implicites pour les systèmes tutoriels intelligents.

Kantharaju, P., Alderfer, K., Zhu, J., Char, B., Smith, B. et Ontanón, S. (2018). Tracing player knowledge in a parallel programming educational game. Dans *Fourteenth Artificial Intelligence and Interactive Digital Entertainment Conference*.

Karagiannidis, C. et Sampson, D. (2004). Adaptation rules relating learning styles research and learning objects meta-data. Dans *Workshop on Individual Differences in Adaptive Hypermedia. 3rd International Conference on Adaptive Hypermedia and Adaptive Web-based Systems (AH2004), Eindhoven, Netherlands*. Citeseer.

Kasurinen, J. et Nikula, U. (2009). Estimating programming knowledge with bayesian knowledge tracing. Dans *ACM SIGCSE Bulletin*, volume 41, 313–317. ACM.

Kennedy, W. G. et Krueger, F. (2013). Building a cognitive model of social trust within act-r. Dans *2013 AAAI Spring Symposium Series*.

Keshtkar, F., Burkett, C., Li, H. et Graesser, A. C. (2014). Using data mining techniques to detect the personality of players in an educational game. In *Educational Data Mining* 125–150. Springer.

Keskar, N. S. et Socher, R. (2017). Improving generalization performance by switching from adam to sgd. *arXiv preprint arXiv :1712.07628*.

Khajah, M., Lindsey, R. V. et Mozer, M. C. (2016). How deep is knowledge tracing ? *arXiv preprint arXiv :1604.02416*.

Kim, H.-N., Alkhaldi, A., El Saddik, A. et Jo, G.-S. (2011). Collaborative user modeling with user-generated tags for social recommender systems. *Expert Systems with Applications*, *38*(7), 8488–8496.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv :1408.5882*.

Kingma, D. P. et Ba, J. (2014). Adam : A method for stochastic optimization. *arXiv preprint arXiv :1412.6980*.

Kobsa, A. (1993). User modeling : Recent work, prospects and hazards. *Human Factors in Information Technology*, *10*, 111–111.

Kobsa, A. (2001). Generic user modeling systems. *User modeling and user-adapted interaction*, *11*(1-2), 49–63.

Kort, B., Reilly, R. et Picard, R. W. (2001). An affective model of interplay between emotions and learning : Reengineering educational pedagogy-building a learning companion. Dans *Proceedings IEEE International Conference on Advanced Learning Technologies*, 43–46. IEEE.

Krizhevsky, A., Sutskever, I. et Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Dans *Advances in neural information processing systems*, 1097–1105.

Kujala, J. V., Richardson, U. et Lyytinen, H. (2010). A bayesian-optimal principle for learner-friendly adaptation in learning games. *Journal of Mathematical Psychology*, *54*(2), 247–255.

Kusner, M., Sun, Y., Kolkin, N. et Weinberger, K. (2015). From word embeddings to document distances. Dans *International Conference on Machine Learning*, 957–966.

Lazarus, A. A. (1973). Multimodal behavior therapy : Treating the" basic id.". *Journal of Nervous and Mental Disease*.

LeCun, Y., Bengio, Y. *et al.* (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, *3361*(10), 1995.

LeCun, Y., Bengio, Y. et Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. et Jackel, L. D. (1989). Backpropagation applied to handwritten zip code

recognition. *Neural computation*, *1*(4), 541–551.

LeCun, Y., Simard, P. Y. et Pearlmutter, B. (1993). Automatic learning rate maximization by on-line estimation of the hessian's eigenvectors. Dans *Advances in neural information processing systems*, 156–163.

Levinson, S. C. (2006). On the human 'interactional engine'.

Lim, M. Y., Dias, J., Aylett, R. et Paiva, A. (2012). Creating adaptive affective autonomous npcs. *Autonomous Agents and Multi-Agent Systems*, *24*(2), 287–311.

Loewenstein, J. et Heath, C. (2009). The repetition-break plot structure : A cognitive influence on selection in the marketplace of ideas. *Cognitive Science*, *33*(1), 1–19.

Lopes, R. et Bidarra, R. (2011). Adaptivity challenges in games and simulations : a survey. *IEEE Transactions on Computational Intelligence and AI in Games*, *3*(2), 85–99.

Loshchilov, I. et Hutter, F. (2017). Fixing weight decay regularization in adam. *arXiv preprint arXiv :1711.05101*.

Lu, H., Setiono, R. et Liu, H. (1996). Effective data mining using neural networks. *IEEE transactions on knowledge and data engineering*, *8*(6), 957–961.

Lu, H., Setiono, R. et Liu, H. (2017). Neurorule : A connectionist approach to data mining. *arXiv preprint arXiv :1701.01358*.

Luo, L., Yin, H., Cai, W., Lees, M., Othman, N. B., Zhou et Suiping (2014). Towards a data-driven approach to scenario generation for serious games. *Computer Animation and Virtual Worlds*, *25*(3-4), 393–402.

Luong, M.-T., Pham, H. et Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv :1508.04025*.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y. et Potts, C. (2011). Learning word vectors for sentiment analysis. Dans *Proceedings of the 49th annual meeting of the association for computational linguistics : Human language technologies-volume 1*, 142–150. Association for Computational Linguistics.

Macedo, L. et Cardoso, A. (2012). The exploration of unknown environments populated with entities by a surprise–curiosity-based agent. *Cognitive*

Systems Research, *19*, 62–87.

Macedo, L., Cardoso, A., Reisenzein, R. et Lorini, E. (2009). Artificial surprise. In *Handbook of research on synthetic emotions and sociable robotics : New applications in affective computing and artificial intelligence* 267–291. IGI Global.

Mairesse, F., Walker, M. A., Mehl, M. R. et Moore, R. K. (2007). Using linguistic cues for the automatic recognition of personality in conversation and text. *Journal of artificial intelligence research*, *30*, 457–500.

Martin, T. G., Burgman, M. A., Fidler, F., Kuhnert, P. M., Low-Choy, S., McBride, M. et Mengersen, K. (2012). Eliciting expert knowledge in conservation science. *Conservation Biology*, *26*(1), 29–38.

Martins, C., Faria, L., De Carvalho, C. V. et Carrapatoso, E. (2008). User modeling in adaptive hypermedia educational systems. *Educational Technology & Society*, *11*(1), 194–207.

Mayo, S. et Sheppard, S. (2001). Housing supply and the effects of stochastic development control. *Journal of Housing Economics*, *10*(2), 109–128.

McCrickard, D. S. et Chewar, C. M. (2003). Attuning notification design to user goals and attention costs. *Communications of the ACM*, *46*(3), 67–72.

McLachlan, G. (2004). *Discriminant analysis and statistical pattern recognition*, volume 544. John Wiley & Sons.

Meyer, D. E. et Kieras, D. E. (1997). A computational theory of executive cognitive processes and multiple-task performance : Part i. basic mechanisms. *Psychological review*, *104*(1), 3.

Michael, D. R. et Chen, S. L. (2005). *Serious games : Games that educate, train, and inform*. Muska & Lipman/Premier-Trade.

Michel, F. et Matthes, F. (2018). A holistic model-based adaptive case management approach for healthcare. Dans *2018 IEEE 22nd International Enterprise Distributed Object Computing Workshop (EDOCW)*, 17–26. IEEE.

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

Min, W., Baikadi, A., Mott, B., Rowe, J., Liu, B., Ha, E. Y. et Lester, J. (2016). A generalized multidimensional evaluation framework for player goal recognition. Dans *Twelfth Artificial Intelligence and Interactive Digital*

Entertainment Conference.

Missura, O. et Gärtner, T. (2009). Player modeling for intelligent difficulty adjustment. Dans *International Conference on Discovery Science*, 197–211. Springer.

Montero, S., Arora, A., Kelly, S., Milne, B. et Mozer, M. (2018). Does deep knowledge tracing model interactions among skills ?. *International Educational Data Mining Society*.

Monterrat, B., Desmarais, M., Lavoué, E. et George, S. (2015). A player model for adaptive gamification in learning environments. Dans *International conference on artificial intelligence in education*, 297–306. Springer.

Munnich, E., Ranney, M. A. et Song, M. (2007). Surprise, surprise : The role of surprising numerical feedback in belief change. Dans *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 29.

Muñoz, K., Mc Kevitt, P., Lunney, T., Noguez, J. et Neri, L. (2011). An emotional student model for game-play adaptation. *Entertainment Computing*, *2*(2), 133–141.

Murdoch, W. J. et Szlam, A. (2017). Automatic rule extraction from long short term memory networks. *arXiv preprint arXiv :1702.02540*.

Musto, C., Semeraro, G., Lovascio, C., de Gemmis, M. et Lops, P. (2018). A framework for holistic user modeling merging heterogeneous digital footprints. Dans *Adjunct Publication of the 26th Conference on User Modeling, Adaptation and Personalization*, 97–101. ACM.

Muyuan, W., Naiyao, Z. et Hancheng, Z. (2004). User-adaptive music emotion recognition. Dans *Proceedings 7th International Conference on Signal Processing, 2004. Proceedings. ICSP'04. 2004.*, volume 2, 1352–1355. IEEE.

Nacke, L. E., Bateman, C. et Mandryk, R. L. (2014). Brainhex : A neurobiological gamer typology survey. *Entertainment computing*, *5*(1), 55–62.

Nesterov, Y. (1983). A method of solving a convex programming problem with convergence rate o (1/k2). Dans *Soviet Mathematics Doklady*, volume 27, 372–376.

Nguyen, T. D., Tran, T., Phung, D. et Venkatesh, S. (2016). Graph-induced restricted boltzmann machines for document modeling. *Information Sciences*, *328*, 60–75.

Nkambou, R., Brisson, J., Kenfack, C., Robert, S., Kissok, P. et Tato, A.

(2015). Towards an intelligent tutoring system for logical reasoning in multiple contexts. In *Design for Teaching and Learning in a Networked World* 460–466. Springer.

Nkambou, R., Mizoguchi, R. et Bourdeau, J. (2010). *Advances in intelligent tutoring systems*, volume 308. Springer Science & Business Media.

Noldus (2014). Facereader : Tool for automaticanalysis of facial expression (version 6.0) [computer software]. Wageningen, the Netherlands :Author.

Nyamen Tato, A. A. (2016). Développement d'un système tutoriel intelligent pour l'apprentissage du raisonnement logique.

Oquab, M., Bottou, L., Laptev, I. et Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. Dans *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1717–1724.

Ortigosa, A., Carro, R. M. et Quiroga, J. I. (2014). Predicting user personality by mining social interactions in facebook. *Journal of computer and System Sciences*, *80*(1), 57–71.

Ortony, A., Clore, G. L. et Collins, A. (1988). The cognitive structure of emotions. *Cambridge Uni.*

Ortony, A., Clore, G. L. et Collins, A. (1990). *The cognitive structure of emotions*. Cambridge university press.

Oudeyer, P.-Y., Gottlieb, J. et Lopes, M. (2016). Intrinsic motivation, curiosity, and learning : Theory and applications in educational technologies. In *Progress in brain research*, volume 229 257–284. Elsevier.

Ouellet, S. (2016). Environnement d'adaptation pour un jeu sérieux.

Paramythis, A., Weibelzahl, S. et Masthoff, J. (2010). Layered evaluation of interactive adaptive systems : framework and formative methods. *User Modeling and User-Adapted Interaction*, *20*(5), 383–453.

Pardos, Z. A. et Heffernan, N. T. (2010). Modeling individualization in a bayesian networks implementation of knowledge tracing. Dans *International Conference on User Modeling, Adaptation, and Personalization*, 255–266. Springer.

Peffers, K., Tuunanen, T., Rothenberger, M. A. et Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of management information systems*, *24*(3), 45–77.

Pekrun, R., Goetz, T., Titz, W. et Perry, R. P. (2002). Academic emotions in students' self-regulated learning and achievement : A program of qualitative and quantitative research. *Educational psychologist*, *37*(2), 91–105.

Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L. J. et Sohl-Dickstein, J. (2015). Deep knowledge tracing. Dans *Advances in neural information processing systems*, 505–513.

Poeller, S., Birk, M. V., Baumann, N. et Mandryk, R. L. (2018). Let me be implicit : Using motive disposition theory to predict and explain behaviour in digital games. Dans *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, p. 190. ACM.

Polyak, B. T. (1964). Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, *4*(5), 1–17.

Popescu, E., Badica, C. et Moraret, L. (2010). Accommodating learning styles in an adaptive educational system. *Informatica*, *34*(4).

Poria, S., Cambria, E., Hussain, A. et Huang, G.-B. (2015). Towards an intelligent framework for multimodal affective data analysis. *Neural Networks*, *63*, 104–116.

Posner, J., Russell, J. A. et Peterson, B. S. (2005). The circumplex model of affect : An integrative approach to affective neuroscience, cognitive development, and psychopathology. *Development and psychopathology*, *17*(3), 715–734.

Qian, Y., Zhang, Y., Ma, X., Yu, H. et Peng, L. (2019). Ears : Emotion-aware recommender system based on hybrid information fusion. *Information Fusion*, *46*, 141–146.

Razmerita, L., Angehrn, A. et Maedche, A. (2003). Ontology-based user modeling for knowledge management systems. Dans *International Conference on User Modeling*, 213–217. Springer.

Reddi, S. J., Kale, S. et Kumar, S. (2018). On the convergence of adam and beyond.

Reddi, S. J., Kale, S. et Kumar, S. (2019). On the convergence of adam and beyond. *arXiv preprint arXiv :1904.09237*.

Reichart, B., Ismailovic, D., Pagano, D. et Brügge, B. (2015). Adaptive serious games. *Comput. Games Softw. Eng*, *9*, 133–149.

Reiser, B. J., Anderson, J. R. et Farrell, R. G. (1985). Dynamic student modelling in an intelligent tutor for lisp programming. Dans *IJCAI*, volume 85, 8–14.

Rennie, J. D. (2005). Regularized logistic regression is strictly convex. *Unpublished manuscript. URL people. csail. mit. edu/jrennie/writing/convexLR. pdf*, *745*.

Ricci, F., Rokach, L. et Shapira, B. (2015). Recommender systems : introduction and challenges. In *Recommender systems handbook* 1–34. Springer.

Rich, E. (1979). User modeling via stereotypes. *Cognitive science*, *3*(4), 329–354.

Rich, E. (1989). Stereotypes and user modeling. In *User models in dialog systems* 35–51. Springer.

Rowe, J., Mott, B., McQuiggan, S., Robison, J., Lee, S. et Lester, J. (2009). Crystal island : A narrative-centered learning environment for eighth grade microbiology. Dans *workshop on intelligent educational games at the 14th international conference on artificial intelligence in education, Brighton, UK*, 11–20.

Rowe, J. P. et Lester, J. C. (2010). Modeling user knowledge with dynamic bayesian networks in interactive narrative environments. Dans *Sixth Artificial Intelligence and Interactive Digital Entertainment Conference*.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv :1609.04747*.

Ruiz, M. d. P. P., Díaz, M. J. F., Soler, F. O. et Pérez, J. R. P. (2008). Adaptation in current e-learning systems. *Computer Standards & Interfaces*, *30*(1-2), 62–70.

Russell, J. A. (2003). Core affect and the psychological construction of emotion. *Psychological review*, *110*(1), 145.

Russell, S. J. et Norvig, P. (2016). *Artificial intelligence : a modern approach*. Malaysia ; Pearson Education Limited,.

Sabourin, J. L. et Lester, J. C. (2013). Affect and engagement in game-basedlearning environments. *IEEE Transactions on Affective Computing*, *5*(1), 45–56.

Saklofske, D. H., Austin, E. J., Mastoras, S. M., Beaton, L. et Osborne, S. E.

(2012). Relationships of personality, affect, emotional intelligence and coping with student stress and academic success : Different patterns of association for stress and success. *Learning and Individual Differences*, *22*(2), 251–257.

Santos, O. C. (2016). Emotions and personality in adaptive e-learning systems : an affective computing perspective. In *Emotions and personality in personalized services* 263–285. Springer.

Savran, A., Gur, R. et Verma, R. (2013). Automatic detection of emotion valence on faces using consumer depth cameras. Dans *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 75–82.

Schaul, T., Zhang, S. et LeCun, Y. (2013). No more pesky learning rates. Dans *International Conference on Machine Learning*, 343–351.

Scherer, K. R. *et al.* (1984). On the nature and function of emotion : A component process approach. *Approaches to emotion*, *2293*, 317.

Shaker, N., Yannakakis, G. et Togelius, J. (2010). Towards automatic personalized content generation for platform games. Dans *Sixth Artificial Intelligence and Interactive Digital Entertainment Conference*.

Shen, L., Wang, M. et Shen, R. (2009). Affective e-learning : Using "emotional" data to improve learning in pervasive learning environment. *Journal of Educational Technology & Society*, *12*(2), 176–189.

Shen, W., Wang, X., Wang, Y., Bai, X. et Zhang, Z. (2015). Deepcontour : A deep convolutional feature learned by positive-sharing loss for contour detection. Dans *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3982–3991.

Sidney, K. D., Craig, S. D., Gholson, B., Franklin, S., Picard, R. et Graesser, A. C. (2005). Integrating affect sensors in an intelligent tutoring system. Dans *Affective Interactions : The Computer in the Affective Loop Workshop at*, 7–13.

Sleeman, D. et Brown, J. S. (1982). *Intelligent tutoring systems*. London : Academic Press.

Smart, P. R., Scutt, T., Sycara, K. et Shadbolt, N. R. (2016). Integrating act-r cognitive models with the unity game engine. In *Integrating Cognitive Architectures into Virtual Character Design* 35–64. IGI Global.

Sreedharan, S., Srivastava, S. et Kambhampati, S. (2018). Hierarchical expertise level modeling for user specific contrastive explanations. Dans *IJCAI*, 4829–4836.

Srivastava, N. et Salakhutdinov, R. R. (2012). Multimodal learning with deep boltzmann machines. Dans *Advances in neural information processing systems*, 2222–2230.

Srivastava, N., Salakhutdinov, R. R. et Hinton, G. E. (2013). Modeling documents with deep boltzmann machines. *arXiv preprint arXiv :1309.6865*.

Starks, K. (2014). Cognitive behavioral game design : a unified model for designing serious games. *Frontiers in psychology*, *5*, 28.

Stern, M., Beck, J. et Woolf, B. P. (1999). Naive bayes classifiers for user modeling. *Center for Knowledge Communication, Computer Science Department, University of Massachusetts*.

Sutskever, I., Martens, J., Dahl, G. et Hinton, G. (2013). On the importance of initialization and momentum in deep learning. Dans *International conference on machine learning*, 1139–1147.

Tadlaoui, M. A., Aammou, S., Khaldi, M. et Carvalho, R. N. (2016). Learner modeling in adaptive educational systems : A comparative study. *International Journal of Modern Education and Computer Science*, *8*(3), 1.

Taherkhani, A., Cosma, G. et McGinnity, T. M. (2018). Deep-fs : A feature selection algorithm for deep boltzmann machines. *Neurocomputing*, *322*, 22–37.

Tai, K. S., Socher, R. et Manning, C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv :1503.00075*.

Tang, D., Qin, B., Liu, T. et Yang, Y. (2015). User modeling with neural network for review rating prediction. Dans *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Tato, A. (2016). Développement d'un système tutoriel intelligent pour l'apprentissage du raisonnement logique.

Tato, A., Nkambou, R., Brisson, J., Kenfack, C., Robert, S. et Kissok, P. (2016). A bayesian network for the cognitive diagnosis of deductive reasoning. Dans *European Conference on Technology Enhanced Learning*, 627–631. Springer.

Tato, A., Nkambou, R., Brisson, J. et Robert, S. (2017a). Predicting learner's deductive reasoning skills using a bayesian network. Dans *International Conference on Artificial Intelligence in Education*, 381–392. Springer.

Tato, A., Nkambou, R. et Dufresne, A. (2019a). Hybrid deep neural networks to predict socio-moral reasoning skills. Dans *Proceedings of the 12th International Conference on Educational Data Mining*, 623–626.

Tato, A., Nkambou, R., Dufresne, A. et Beauchamp, M. H. (2017b). Convolutional neural network for automatic detection of sociomoral reasoning level. 284–289.

Tato, A., Nkambou, R., Dufresne, A. et Frasson, C. (2018a). Semi-supervised multimodal deep learning model for polarity detection in arguments. Dans *2018 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Tato, A., Nkambou, R. et Frasson, C. (2018b). Predicting emotions from multimodal users' data. Dans *Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization*, 369–370. ACM.

Tato, A., Nkambou, R. et Ghali, R. (2019b). Towards predicting attention and workload during math problem solving. Dans *International Conference on Intelligent Tutoring Systems*, 224–229. Springer.

Terzis, V., Moridis, C. N. et Economides, A. A. (2010). Measuring instant emotions during a self-assessment test : the use of facereader. Dans *Proceedings of the 7th International Conference on Methods and Techniques in Behavioral Research*, p. 18. ACM.

Tondello, G. F., Wehbe, R. R., Diamond, L., Busch, M., Marczewski, A. et Nacke, L. E. (2016). The gamification user types hexad scale. Dans *Proceedings of the 2016 annual symposium on computer-human interaction in play*, 229–243. ACM.

Towell, G. G. et Shavlik, J. W. (1994). Knowledge-based artificial neural networks. *Artificial intelligence*, *70*(1-2), 119–165.

Truong, H. M. (2016). Integrating learning styles and adaptive e-learning system : Current developments, problems and opportunities. *Computers in human behavior*, *55*, 1185–1193.

Tsamardinos, I., Brown, L. E. et Aliferis, C. F. (2006). The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, *65*(1), 31–78.

Tyng, C. M., Amin, H. U., Saad, M. N. et Malik, A. S. (2017). The influences of emotion on learning and memory. *Frontiers in psychology*, *8*, 1454.

Um, E., Plass, J. L., Hayward, E. O., Homer, B. D. *et al.* (2012). Emotional design in multimedia learning. *Journal of educational psychology*, *104*(2), 485.

Vachiratamporn, V., Moriyama, K., Fukui, K.-i. et Numao, M. (2014). An implementation of affective adaptation in survival horror games. Dans *2014 IEEE Conference on Computational Intelligence and Games*, 1–8. IEEE.

Van Melle, W. (1978). Mycin : a knowledge-based consultation program for infectious disease diagnosis. *International journal of man-machine studies*, *10*(3), 313–322.

Vanlehn, K. (2006). The behavior of tutoring systems. *International journal of artificial intelligence in education*, *16*(3), 227–265.

Wang, L., Sy, A., Liu, L. et Piech, C. (2017). Deep knowledge tracing on programming exercises. Dans *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale*, 201–204. ACM.

Wang, S., Yang, Y., Culpepper, S. A. et Douglas, J. A. (2018). Tracking skill acquisition with cognitive diagnosis models : A higher-order, hidden markov model with covariates. *Journal of Educational and Behavioral Statistics*, *43*(1), 57–87.

Werbos, P. J. *et al.* (1990). Backpropagation through time : what it does and how to do it. *Proceedings of the IEEE*, *78*(10), 1550–1560.

Wöllmer, M., Kaiser, M., Eyben, F., Schuller, B. et Rigoll, G. (2013). Lstm-modeling of continuous emotions in an audiovisual affect recognition framework. *Image and Vision Computing*, *31*(2), 153–163.

Woolf, B. P. (2010). *Building intelligent interactive tutors : Student-centered strategies for revolutionizing e-learning*. Morgan Kaufmann.

Wouters, P., Van Nimwegen, C., Van Oostendorp, H. et Van Der Spek, E. D. (2013). A meta-analysis of the cognitive and motivational effects of serious games. *Journal of educational psychology*, *105*(2), 249.

Wu, C., Wu, F., Liu, J. et Huang, Y. (2019). Hierarchical user and item representation with three-tier attention for recommendation. Dans *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers)*, 1818–1826.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R. et Bengio, Y. (2015). Show, attend and tell : Neural image caption generation with visual attention. Dans *International Conference on Machine Learning*,

2048–2057.

Yannakakis, G. N. et Hallam, J. (2009). Real-time game adaptation for optimizing player satisfaction. *IEEE Transactions on Computational Intelligence and AI in Games*, *1*(2), 121–133.

Yannakakis, G. N., Spronck, P., Loiacono, D. et André, E. (2013). Player modeling. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Yannakakis, G. N., Togelius, J., Khaled, R., Jhala, A., Karpouzis, K., Paiva, A. et Vasalou, A. (2010). Siren : Towards adaptive serious games for teaching conflict resolution. *Proceedings of ECGBL*, 412–417.

Yosinski, J., Clune, J., Bengio, Y. et Lipson, H. (2014). How transferable are features in deep neural networks ? Dans *Advances in neural information processing systems*, 3320–3328.

Yudelson, M. V., Koedinger, K. R. et Gordon, G. J. (2013). Individualized bayesian knowledge tracing models. Dans *International Conference on Artificial Intelligence in Education*, 171–180. Springer.

Zakrzewska, D. (2008). Cluster analysis for users' modeling in intelligent e-learning systems. Dans *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, 209–214. Springer.

Zappone, A., Di Renzo, M., Debbah, M., Lam, T. T. et Qian, X. (2018). Model-aided wireless artificial intelligence : Embedding expert knowledge in deep neural networks towards wireless systems optimization. *arXiv preprint arXiv :1808.01672*.

Zeiler, M. D. (2012). Adadelta : an adaptive learning rate method. *arXiv preprint arXiv :1212.5701*.

Zeng, Z., Tu, J., Liu, M., Huang, T. S., Pianfetti, B., Roth, D. et Levinson, S. (2007). Audio-visual affect recognition. *IEEE Transactions on multimedia*, *9*(2), 424–428.

Zhang, H., Song, Y. et Song, H.-T. (2007). Construction of ontology-based user model for web personalization. Dans *International Conference on User Modeling*, 67–76. Springer.

Zhang, J., Shi, X., King, I. et Yeung, D.-Y. (2017a). Dynamic key-value memory networks for knowledge tracing. Dans *Proceedings of the 26th international conference on World Wide Web*, 765–774. International World Wide Web Conferences Steering Committee.

Zhang, K. et Yao, Y. (2018). A three learning states bayesian knowledge tracing model. *Knowledge-Based Systems*, *148*, 189–201.

Zhang, L., Xiong, X., Zhao, S., Botelho, A. et Heffernan, N. T. (2017b). Incorporating rich features into deep knowledge tracing. Dans *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale*, 169–172. ACM.

Zhu, Y., Li, H., Liao, Y., Wang, B., Guan, Z., Liu, H. et Cai, D. (2017). What to do next : Modeling user behaviors by time-lstm. Dans *IJCAI*, 3602–3608.

Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. Dans *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, 928–936.