THESE DE DOCTORAT DE L'ETABLISSEMENT UNIVERSITE BOURGOGNE FRANCHE-COMTE

PREPAREE A L'UNIVERSITE DE FRANCHE-COMTE

Ecole doctorale nº <numéro de l'Ecole doctorale>

<Nom de l'Ecole doctorale>

Doctorat de Informatique

Par

M. SOTO FORERO Daniel

Adaptation en temps réel d'une séance d'entraînement par intelligence artificielle

Thèse présentée et soutenue à Besançon, le <date>

Composition du Jury :

<Civilité><Nom><Prénom>	<Fonction et établissement d'exercice>	Président
<Civilité><Nom><Prénom>	<Fonction et établissement d'exercice>	Rapporteur
<Civilité><Nom><Prénom>	<Fonction et établissement d'exercice>	Rapporteur
<Civilité><Nom><Prénom>	<Fonction et établissement d'exercice>	Examinateur
<Civilité><Nom><Prénom>	<Fonction et établissement d'exercice>	Examinatrice
<Civilité><Nom><Prénom>	<Fonction et établissement d'exercice>	Directeur de thèse
<Civilité><Nom><Prénom>	<Fonction et établissement d'exercice>	Codirecteur de thèse
<Civilité><Nom><Prénom>	<Fonction et établissement d'exercice>	Invité

REMERCIEMENTS

SOMMAIRE

I

CONTEXTE ET PROBLÉMATIQUES

1

INTRODUCTION

Ce chapitre introductif vise à expliquer les termes du sujet et à introduire la thématique principale abordée. Il présente la problématique de cette thèse, introduit les différentes contributions proposées et annonce le plan du manuscrit.

Comme l'indique [Nkambou et al., 2010], les environnements informatiques pour l'apprentissage humain (EIAH) sont des outils proposant des services devant permettre aux apprenants d'acquérir des connaissances et de développer des compétences dans un domaine spécifique. Pour fournir des services efficaces, le système doit intégrer une représentation des connaissances du domaine et des mécanismes pour utiliser ces connaissances. Il doit également être en mesure de raisonner et de résoudre des problèmes.

Le système *Artificial Intelligence - Virtual Trainer* (AI-VT) est un EIAH générique développé au département d'informatique des systèmes complexes (DISC) de l'institut de recherche FEMTO-ST. Cet outil informatique propose un ensemble d'exercices aux apprenants dans le cadre de séances d'entrainement. AI-VT intègre le fait qu'une séance d'entraînement se situe dans un cycle de plusieurs séances. Les réponses apportées par l'apprenant à chaque exercice sont évaluées numériquement sur une échelle prédéfinie, ce qui permet d'estimer les progrès de l'apprenant et de déduire les sous-domaines dans lesquels il peut avoir des difficultés. Une séance est générée par un système multi-agents associé à un système de raisonnement à partir de cas (RàPC) [Henriet et al., 2017]. Un apprenant choisit le domaine dans lequel il souhaite s'entraîner et AI-VT lui propose un test préliminaire. Les résultats obtenus permettent de placer l'apprenant dans le niveau de maîtrise adéquate. Le système génère ensuite une séance adaptée veillant à l'équilibre entre l'entraînement, l'apprentissage et la découverte de nouvelles connaissances. L'actualisation du niveau de l'apprenant est effectuée à la fin de chaque séance. De cette façon l'apprenant peut avancer dans l'acquisition des connaissances ou s'entraîner sur des connaissances déjà apprises.

Un certain nombre d'EIAH utilisent des algorithmes d'intelligence artificielle (IA) pour détecter les faiblesses et aussi pour s'adapter à chaque apprenant. Les algorithmes et modèles de certains de ces systèmes seront analysés dans les chapitres 3 et 4 . Ces chapitres présenteront leurs propriétés, leurs avantages et leurs limites.

Le système AI-VT initial était capable d'ajuster les paramètres de personnalisation d'une séance à l'autre, mais il ne pouvait pas modifier une séance en cours même si certains exercices de celle-ci était trop simples ou trop complexes. Chaque séance était figée et devait être déroulée jusqu'à son terme avant de pouvoir identifier des acquis et des lacunes. Les travaux de cette thèse ont eu pour objectif de pallier ce manque.

La problématique principale de cette thèse est la personnalisation en temps réel du parcours d'apprentissage des apprenants dans le système AI-VT (*Artificial Intelligence - Virtual Trainer*)

Ici *le temps réel* est considéré comme étant le moment où se déroule la séance que l'apprenant est en train de suivre. Par conséquent, l'objectif est de rendre AI-VT plus dynamique pour l'identification des difficultés et l'adaptation du contenu personnalisé en fonction des connaissances démontrées.

La partie suivante présente une liste des principales contributions apportées par cette thèse à la problématique générale énoncée plus haut.

Ce travail de thèse a été éffectué au sein l'Université de Franche-Comté (UFC) devenue depuis le 1er janvier 2025 l'Université Marie et Louis Pasteur (UMLP). Ces recherches ont été menées au sein de l'équipe DEODIS du département d'informatique des systèmes complexes de l'institut de recherche FEMTO-ST, unité mixte de recherche (UMR) 6174 du centre national de la recherche scientifique (CNRS).

1.1/ CONTRIBUTIONS PRINCIPALES

La problématique principale de ces travaux de recherche a été déclinée en plusieurs sous-parties. Ces dernières sont présentées ci-dessous sous la forme de questions. Pour chacune d'elles, une ou plusieurs propositions ont été faites. Voici les questions de recherche abordées et les contributions apportées :

1. **Comment permettre au système AI-VT d'évoluer et d'intégrer de multiples outils ?** Pour répondre à cette question, une architecture modulaire autour du moteur initial d'AI-VT a été conçue et implémentée (ce moteur initial étant le système de raisonnement à partir de cas (RàPC) développé avant le démarrage de ces travaux de thèse).
2. **Comment déceler qu'un exercice est plus ou moins adapté aux besoins de l'apprenant ?** Choisir les exercices les plus adaptés nécessite d'associer une valeur liée à son utilité pour cet apprenant. Les modèles de régression permettent d'interpoler une telle valeur. Pour répondre à cette question, nous avons proposé de nouveaux modèles de régression fondés sur le paradigme du raisonnement à partir de cas et celui des systèmes multi-agents.
3. **Quel modèle et quel type d'algorithmes peuvent être utilisés pour recommander un parcours personnalisé aux apprenants ?** Pour apporter une réponse à cette question, un système de recommandation fondé sur l'apprentissage par renforcement a été conçu. L'objectif de ces travaux est de proposer un module permettant de recommander des exercices aux apprenants en fonction des connaissances demontrées et en se fondant sur les réponses apportées aux exercices précédents de la séance en cours. Ce module de révision de la séance en cours du modèle de RàPC est fondé sur un modèle Bayésien.
4. **Comment consolider les acquis de manière automatique ?** Une séance doit non seulement intégrer des exercices de niveaux variés mais également permettre à l'apprenant de renforcer ses connaissances. Dans cette optique, notre modèle Bayésien a été enrichi d'un processus de Hawkes incluant une fonction d'oubli.

1.2/ PLAN DE LA THÈSE

Ce manuscrit est scindé en deux grandes parties. La première partie contient trois chapitres et la seconde en contient quatre. Le premier chapitre de la première partie (chapitre 2) vise à introduire le sujet, les concepts, les algorithmes associés, présenter le contexte général et l'application AI-VT initiale. Le deuxième chapitre de cette partie présente différents travaux emblématiques menés dans le domaine des environnements informatiques pour l'apprentissage humain. Le chapitre suivant conclut cette première partie en présentant le raisonnement à partir de cas.

Dans la seconde partie du manuscrit, le chapitre 5 explicite l'architecture proposée pour intégrer les modules développés autour du système AI-VT initial et élargir ses fonctionnalités. Le chapitre 6 propose deux outils originaux fondés sur le raisonnement à partir de cas et les systèmes multi-agents pour résoudre des problèmes de régression de façon générique. Le chapitre 7 présente l'application de ces nouveaux outils de régression dans un système de recommandation intégré à AI-VT utilisant un modèle Bayésien. Ce chapitre montre de quelle manière il permet de réviser une séance d'entraînement en cours. Le chapitre 8 montre l'utilisation du processus de Hawkes pour renforcer l'apprentissage.

Enfin, une conclusion générale incluant des perspectives de recherche termine ce manuscrit.

2

CONTEXTE

Dans ce chapitre sont décrits plus en détails le contexte applicatif et le contexte technique de ces travaux de recherche. Il présente les concepts et algorithmes utilisés dans le développement des modules. Ces modules font partie des contributions de cette thèse à l'environnement informatique pour l'apprentissage humain (EIAH) appelé AI-VT (*Artificial Intelligence - Artificial Trainer*).

Ce chapitre commence par une présentation des EIAH suivie d'une présentation sommaire du fonctionnement d'AI-VT. AI-VT étant un système de raisonnement à partir de cas (RàPC) modélisé sous la forme d'un système multi-agents (SMA), cette présentation des EIAH est suivie d'une présentation du RàPC, puis des SMA. Ce chapitre se termine par une présentation de différents algorithmes et notions implémentés dans AI-VT pour la personnalisation et l'adaptation des séances d'entraînement proposées par l'EIAH.

2.1/ LES STRATÉGIES D'APPRENTISSAGE HUMAIN ET LES ENVIRONNEMENTS INFORMATIQUES POUR L'APPRENTISSAGE HUMAIN (EIAH)

2.1.1/ LES STRATÉGIES D'APPRENTISSAGE

Dans [Lalitha and Sreeja, 2020], l'apprentissage est défini comme une fonction du cerveau humain acquise grâce au changement permanent dans l'obtention de connaissances et de compétences par le biais d'un processus de transformation du comportement lié à l'expérience ou à la pratique. L'apprentissage implique réflexion, compréhension, raisonnement et mise en œuvre. Un individu peut utiliser différentes stratégies pour acquérir la connaissance. Comme le montre la figure 2.1, les stratégies peuvent être classées entre mode traditionnel et mode en-ligne.

L'apprentissage traditionnel se réfère à l'apprentissage-type tel qu'il est fait dans une classe. Il est centré sur l'enseignant où la présence physique est l'élément fondamental dans la mesure où elle implique une unité de temps et de lieu. Ici, l'enseignant interagit directement avec l'élève, et les ressources sont généralement des documents imprimés. L'apprentissage et la participation doivent y être actifs, la rétroaction y est instantanée et il existe des interactions sociales. En revanche, les créneaux horaires, les lieux et les contenus sont rigides (décidés par l'enseignant et la structure dans laquelle les enseignements sont dispensés).

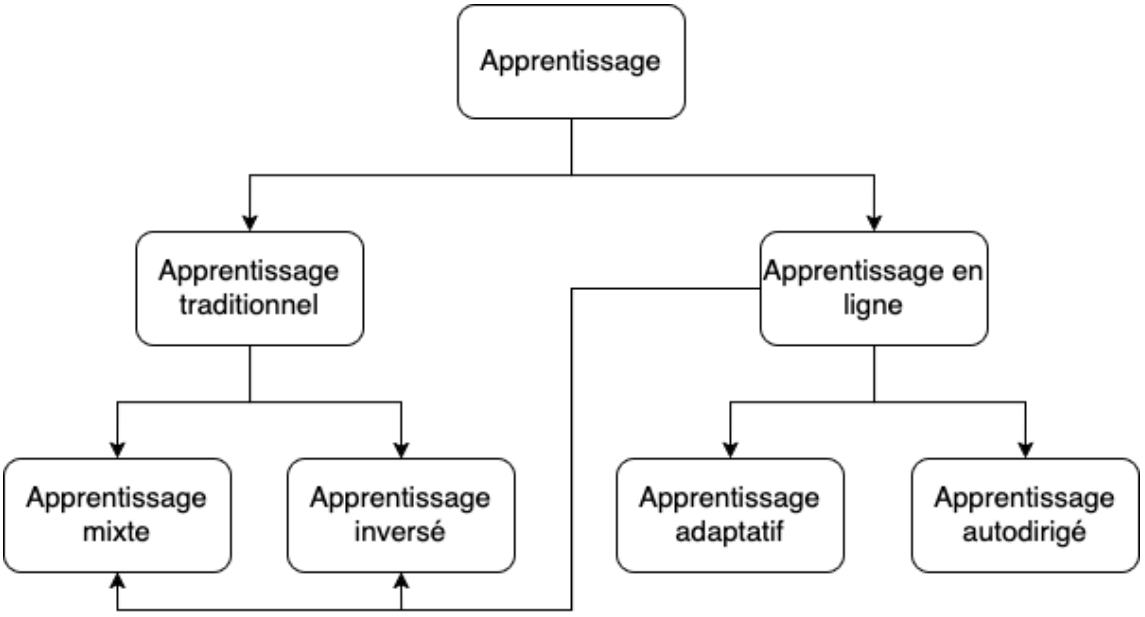

FIGURE 2.1 – Stratégies d'apprentissage (Traduit de [Lalitha and Sreeja, 2020])

L'autre stratégie globale est l'apprentissage en-ligne, qui s'appuie sur les ressources et l'information disponible sur le web. Cette stratégie incite les apprenants à être actifs pour acquérir de nouvelles connaissances de manière autonome grâce aux nombreuses ressources disponibles. Les points positifs de cette stratégie sont par exemple la rentabilité, la mise à niveau continue des compétences et des connaissances ou une plus grande opportunité d'accéder au contenu du monde entier. Pour l'apprenant, l'auto-motivation et l'interaction peuvent être des défis à surmonter. Pour l'enseignant, il est parfois difficile d'évaluer l'évolution du processus d'apprentissage de l'individu. L'apprentissage en-ligne fait aussi référence à l'utilisation de dispositifs électroniques pour apprendre (*e-learning*) où les apprenants peuvent être guidés par l'enseignant à travers des liens, du matériel spécifique d'apprentissage, des activités ou exercices.

Il existe également des stratégies hybrides combinant des caractéristiques des deux stratégies fondamentales, comme un cours en-ligne mais avec quelques séances en présentiel pour des activités spécifiques ou des cours traditionnels avec du matériel d'apprentissage complémentaire en-ligne.

2.1.2/ LES EIAH

Les EIAH sont des outils pour aider les apprenants dans le processus d'apprentissage et l'acquisition de la connaissance. Ces outils sont conçus pour fonctionner avec des stratégies d'apprentissage en-ligne et mixtes. Généralement, l'architecture de ces systèmes est divisée en quatre composants comme le montre la figure 2.2. Ces composants sont :

— Le *domaine* (*Domain Model* sur la figure 2.2), qui contient les concepts, les règles et les stratégies pour résoudre des problèmes dans un domaine spécifique. Ce composant peut détecter et corriger les erreurs des apprenants. En général, l'information est divisée en séquences pédagogiques.
— Le composant *apprenant* (*Student Model* sur la figure 2.2), qui est le noyau du

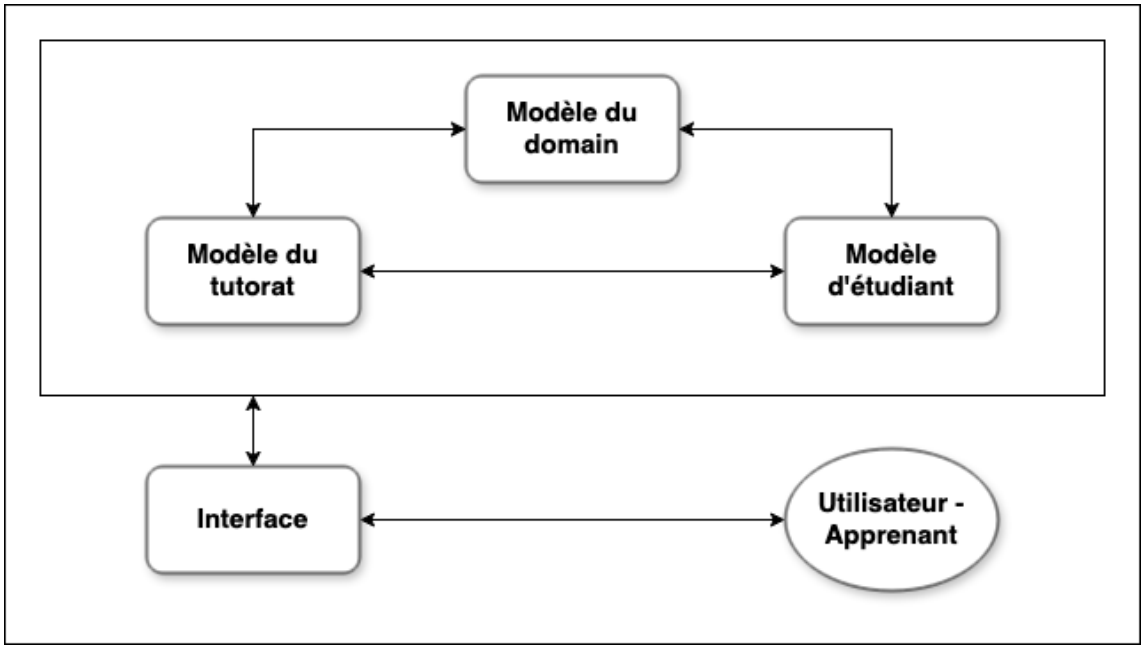

FIGURE 2.2 – L'architecture générale des EIAH, les composantes et leurs interactions (Traduit de [Nkambou et al., 2010])

système car il contient toute l'information utile à l'EIAH concernant l'apprenant (*User-Learner* sur la figure 2.2). Ce composant contient également les informations sur l'évolution de l'acquisition des compétences de l'apprenant. Ce module doit avoir les informations implicites et explicites pour pouvoir créer une représentation des connaissances acquises, non et partiellement acquises et l'évolution de l'apprentissage de l'apprenant. Ces informations doivent permettre de générer un diagnostic de l'état de la connaissance de l'apprenant, à partir duquel le système peut prédire les résultats dans certains domaines et choisir la stratégie optimale pour présenter les nouveaux contenus.

— L'*enseignant* (*Tutoring Model* sur la figure 2.2) est également modélisé sous la forme d'un composant. Il reçoit l'information des modules précédents, information grâce à laquelle il peut prendre des décisions sur le changement de parcours ou sur la stratégie d'apprentissage. Il peut également interagir avec l'apprenant.

— L'*interface* (*Interface* sur la figure 2.2) est le module chargé de la gestion des configurations de l'EIAH et des interactions entre ses composants.

Le développement et la configuration de ce type de systèmes relèvent de multiples disciplines et impliquent plusieurs domaines de recherche parmi lesquels figurent la pédagogie, l'éducation, la psychologie, les sciences cognitives et l'intelligence artificielle.

2.1.3/ L'EXERCISEUR INITIAL AI-VT

Le DISC travaille depuis plusieurs années sur l'utilisation potentielle de l'intelligence artificielle dans le cadre d'un exerciseur couvrant plusieurs domaines d'apprentissage. Cet exerciseur, appelé AI-VT(Artificial Intelligence Virtual Trainer) est fondé sur différents concepts d'intelligence artificielle et ses domaines d'application sont l'apprentissage de l'aïkido, des bases de l'algorithmique et de l'anglais. Un premier réseau de convolution

détermine les lacunes de l'apprenant en analysant les résultats de quelques exercices préliminaires. Puis un système de raisonnement à partir de cas distribué prend le relais et détermine une liste d'exercices à proposer en regard de ses lacunes et en tenant compte des séances qui ont déjà été proposées par le système à cet apprenant.

Dans le système AI-VT, une séance d'entraînement est proposée dans le cadre d'un cycle d'entraînement. Ainsi, un exercice proposé en première séance du cycle peut être reproposé ensuite dans une autre séance, afin de consolider les acquis et de remettre en mémoire certaines connaissances à l'apprenant. Une séance est entièrement consacrée à une seule et même compétence, déclinée en plusieurs sous-compétences. Des exercices dans différentes sous-compétences sont proposées à chaque séance. Les exercices d'une même sous-compétence sont regroupés. Un même exercice pouvant permettre de travailler deux sous-compétences différentes, un mécanisme de règlement des conflits a été implémenté dans AI-VT afin qu'un même exercice ne puisse être proposé plus d'une seule fois dans une même séance d'entraînement. Dans AI-VT, l'énoncé d'un exercice est constitué de deux parties : le contexte et la question. Cette structure modulaire permet d'associer plusieurs questions à un même contexte et inversement. Les réponses apportées par les apprenants à chaque exercice sont notées sur 10 points et le temps mis pour répondre à chaque question est comptabilisée. L'étudiant dispose d'une interface présentant un tableau de bord des exercices, sous-compétences et compétences qu'il ou elle a travaillé.

Pour l'apprentissage de l'Anglais, un robot est chargé d'énoncer les exercices de la séance. Au démarrage de cette thèse, les apprenants ont une liste personnalisée d'exercices proposée par le système AI-VT, mais c'est à l'enseignant de vérifier la validité des algorithmes/solutions proposés par les étudiants et d'identifier leurs difficultés. L'amélioration du système et de la personnalisation du parcours de l'apprenant implique une correction automatique des exercices et cela sans utiliser un entrainement spécifique à chaque exercice, mais cet objectif ne fait pas partie de cette thèse. La définition de profils d'apprenants par le recueil de traces reste également à travailler pour optimiser les aides et séances d'exercices.

Le système AI-VT a été implémenté en se fondant sur deux paradigmes de l'IA : la séance d'entraînement est construite par différents modules suivant la philosophie du raisonnement à partir de cas, et l'architecture logicielle a été modélisée selon un système multi-agents. Une présentation sommaire de ces deux paradigmes de l'IA sont présentés dans cette section, et celles-ci sont suivies de la présentation de différents algorithmes et fonctions implémentés dans l'EIAH AI-VT. Des états de l'art plus complets sur les EIAH et le RàPC sont présentés dans le chapitre suivant. Le système AI-VT, son architecture et les évolutions qui ont été réalisées sur celle-ci sont détaillés dans le chapitre 4.

2.2/ LE CONTEXTE TECHNIQUE

2.2.1/ LE RAISONNEMENT À PARTIR DE CAS (RÀPC)

Le raisonnement à partir de cas est un modèle de raisonnement fondé sur l'hypothèse que les problèmes similaires ont des solutions similaires. Ainsi, les systèmes de RàPC infèrent une solution à un problème posé à partir des solutions mises en œuvre auparavant pour résoudre d'autres problèmes similaires [Roldan Reyes et al., 2015].

Un cas est défini comme étant la représentation d'un problème et la description de sa solution. Les cas résolus sont stockés et permettent au système de RàPC de construire de nouveaux cas à partir de ceux-ci. Formellement, si P est l'espace des problèmes et S l'espace des solutions, alors un problème x et sa solution y appartiennent à ces espaces : $x \in P$ et $y \in S$. Si y est solution de x alors, un cas est représenté par le couple $(x, y) \in P \times S$. Le RàPC a besoin d'une base de N problèmes et de leurs solutions associées. Cette base est appelée base de cas. Ainsi tout cas d'indice $n \in [1, N]$ de cette base de cas est formalisé de la manière suivante : (x^n, y^n). L'objectif du RàPC est, étant donné un nouveau problème x^z de trouver sa solution y^z en utilisant les cas stockés dans la base de cas [Lepage et al., 2020].

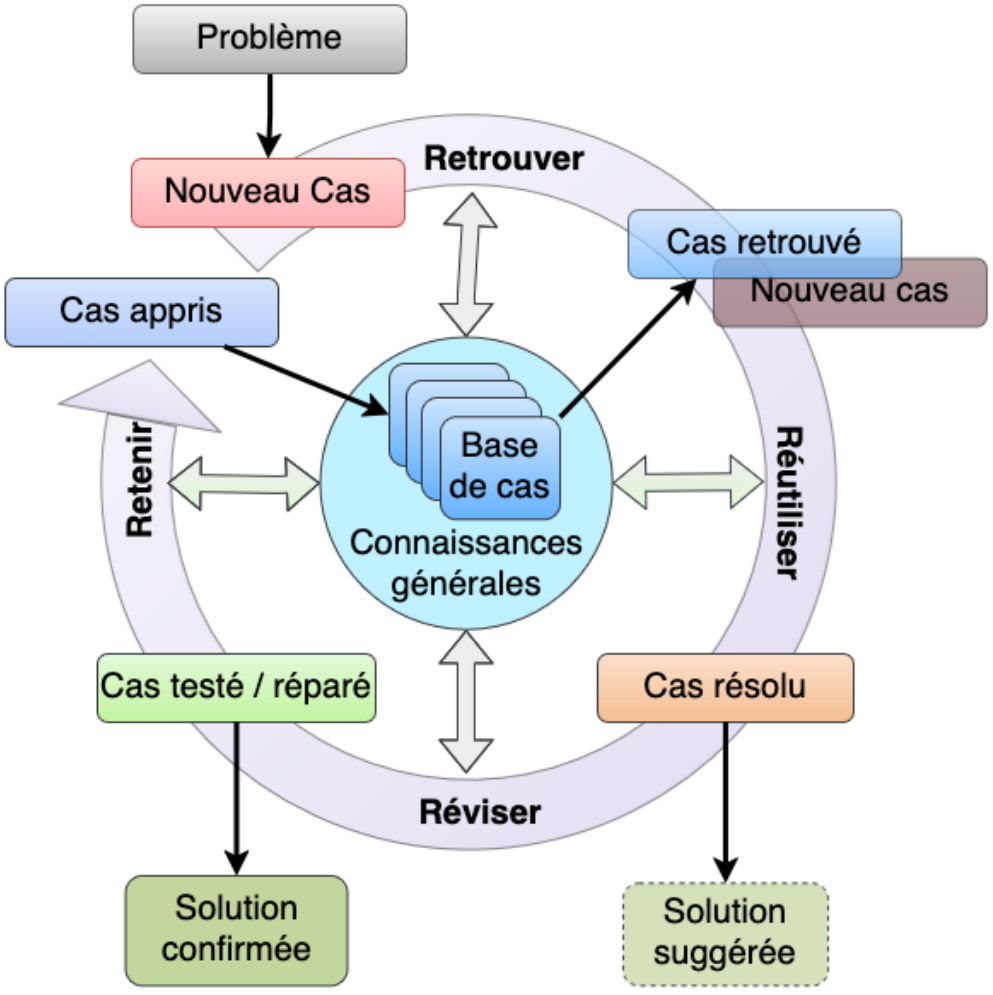

FIGURE 2.3 – Cycle fondamental du raisonnement à partir de cas (Adapté et traduit de [Leikola et al., 2018])

Le processus du RàPC est divisé en quatre étapes formant un cycle comme l'ont proposé Aamont et Plaza [Aamodt and Plaza, 1994]. La figure 2.5 montre le cycle, le flux d'informations ainsi que les relations entre chacune des étapes [Leikola et al., 2018] : re-

trouver (rechercher), réutiliser (adapter), réviser et retenir (stocker). Chacune des étapes est décrite dans [Richter and Weber, 2013] de la manière suivante.

2.2.1.1/ RETROUVER (RECHERCHER)

L'objectif de cette étape est de rechercher dans la base de cas, les cas similaires à un nouveau cas donné. Cette similarité n'est pas un concept général, mais dépend du contexte, de l'objectif et du type de données. La question fondamentale qui se pose dans cette étape est : quel est le cas le plus approprié dans la base de cas qui permet de réutiliser sa solution pour résoudre le nouveau problème donné ?.

Le nouveau cas est comparé à chaque cas de la base afin d'en évaluer la similitude. La comparaison entre les cas est différente selon la structure de la base et la manière dont les problèmes sont décrits dans celle-ci. Généralement, un problème est représenté par un ensemble d'attributs aux valeurs spécifiques. Cette représentation est connue comme la représentation attribut-valeur. La similitude entre deux cas de ce type est calculée suivant l'équation 2.1. La similitude entre deux cas attribut-valeur $c_1 = a_{1,1}, a_{1,1}, ..., a_{1,n}$ et $c_2 = a_{2,1}, a_{2,1}, ..., a_{2,n}$ est la somme pondérée des similitudes entre les attributs considérés individuellement. La pondération détermine l'importance de chaque attribut.

$$\sum_{i=1}^{n} \omega_i \; sim(a_{1,i}, a_{2,i}) \tag{2.1}$$

Dans cette étape, il est possible de sélectionner k différents cas similaires au nouveau cas donné.

2.2.1.2/ RÉUTILISER (ADAPTER)

L'idée du RàPC est d'utiliser l'expérience pour essayer de résoudre de nouveaux cas (problèmes). La figure 2.4 montre le principe de réutilisation d'un cas (problème) résolu (appelé *cas source*) auparavant pour le proposer en solution d'un nouveau cas (dénommé *cas cible*) après adaptation.

Si certains cas cibles sont très similaires au cas source le plus proche, il est cependant souvent nécessaire d'adapter la solution du cas source le plus similaire afin d'arriver à une solution satisfaisante pour le cas cible. De la même manière que pour la phase de recherche du cas le plus similaire, il existe de nombreuses stratégies d'adaptation. Celles-ci dépendent de la représentation des cas, du contexte dans lequel le RàPC est mis en œuvre et des algorithmes utilisés pour l'adaptation. L'une des stratégies les plus utilisées est la définition d'un ensemble de règles transformant la ou les solutions des cas sources les plus similaires au cas cible. L'objectif ici est d'inférer une nouvelle solution y^z du cas x^z à partir des k cas sources retrouvés dans la première étape.

2.2.1.3/ RÉVISER ET RÉPARER

La solution adaptée proposée doit ensuite être évaluée et révisée afin de satisfaire certains critères de validation. L'étape de révision est chargée d'évaluer l'applicabilité de la

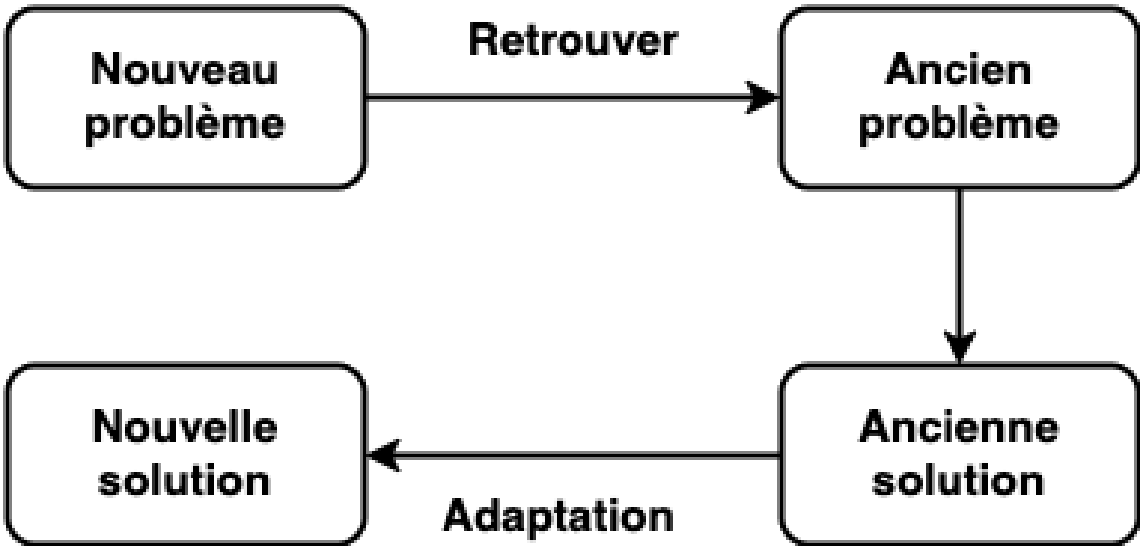

FIGURE 2.4 – Principe de réutilisation dans le RàPC (Traduit de [Richter and Weber, 2013])

solution cible obtenue suite à l'étape 'adaptation'. Cette évaluation peut être réalisée dans le monde réel ou dans une simulation.

L'objectif de cette phase de révision est la validation du couple (x^z, y^z). Autrement dit, le but est ici de vérifier si la solution trouvée y^z résout le problème x^z et satisfait les règles de validité et d'applicabilité. Si la solution n'est pas correcte, l'expert mettant en œuvre la solution doit ajuster ou modifier la solution cible proposée.

Le processus d'évaluation, de révision et de réparation peut être mis en œuvre via un processus d'apprentissage permettant d'améliorer la détection et la correction des failles des futures solutions générées. Dans les travaux de cette thèse, nous avons envisagé la possibilité d'utiliser différents outils d'apprentissage pour évaluer et réviser les solutions cibles dans certains cas particuliers.

L'évaluation peut être faite :

— par un expert humain capable d'évaluer la solution et sa pertinence dans le contexte applicatif,
— par un système fonctionnant en production et renvoyant le résultat de l'application de la solution,
— par un modèle statistique ou un système de test.

La correction des solutions peut être automatique, mais elle dépend du domaine et de l'information disponible. Un ensemble de règles peut aider à identifier les solutions non valides.

2.2.1.4/ RETENIR (STOCKER)

Si le cas cible résolu est jugé pertinent, alors celui-ci peut être stocké dans la base de cas pour aider ensuite à la résolution de futurs cas cible. La décision de stocker ou non un nouveau cas dans la base de cas doit tenir compte de la capacité de cette nouvelle base de cas à proposer de futures solutions pertinentes, évaluer et corriger les solutions

cibles générées d'une part, et maintenir une base de cas d'une taille ne gaspillant pas inutilement des ressources de stockage et de calculs du système de RàPC d'autre part.

2.2.1.5/ CONTENEURS DE CONNAISSANCE

Pour pouvoir exécuter le cycle complet, le RàPC dépend de quatre sources différentes d'information. Ces sources sont appelées les conteneurs de connaissances par [Richter, 2009]. Cet article définit les conteneurs suivants dans les systèmes de RàPC : le conteneur de cas, le conteneur d'adaptation, le conteneur du vocabulaire et le conteneur de similarité :

- Le conteneur de cas contient les expériences passées que le système peut utiliser pour résoudre les nouveaux problèmes. L'information est structurée sous la forme d'un couple (p, s), où p est la description d'un problème et s est la description de sa solution.
- Le conteneur d'adaptation stocke la ou les stratégies d'adaptation ainsi que les règles et paramètres nécessaires pour les exécuter.
- Le conteneur du vocabulaire contient l'information, sa signification et sa terminologie. Les éléments comme les entités, les attributs, les fonctions ou les relations entre les entités peuvent y être décrits.
- Le conteneur de similarité contient l'ensemble des connaissances liées et nécessaires au calcul de la similarité entre deux cas : les fonctions de calcul de similarité et les paramètres de mesure de similarité $sim(p_1, p_2)$ entre les cas p_1 et p_2.

La figure 2.5 montre les flux d'informations entre les étapes du RàPC et les conteneurs. Les flèches continues de la figure 2.5 décrivent l'ordre dans lequel les phases du cycle du RàPC sont exécutées. Les flèches avec des lignes discontinues matérialisent les flux d'information, c'est-à-dire les liens entre les étapes du cycle et les conteneurs de connaissance.

2.2.2/ LES SYSTÈMES MULTI-AGENTS

Les systèmes multi-agents sont des systèmes conçus pour résoudre des problèmes en combinant l'intelligence artificielle et le calcul distribué. Ces systèmes sont composés de multiples entités autonomes appelées agents, qui ont la capacité de communiquer entre elles et également de coordonner leurs comportements [Hajduk et al., 2019].

Les agents ont les propriétés suivantes :

- Autonomie : les agents fonctionnent sans intervention externe des êtres humains ou d'autres entités, ils ont un mécanisme interne qui leur permet de contrôler leurs états.
- Réactivité : les agents ont la capacité de percevoir l'environnement dans lequel ils sont et peuvent réagir aux changements qui se produisent.
- Pro-activité : les agents peuvent effectuer des changements, réagir à différents stimuli provenant de leur environnement et s'engager dans un processus cognitif interne.
- Coopération : les agents peuvent communiquer les uns avec les autres, échanger des informations afin de se coordonner et résoudre un même problème.
- Apprentissage : il est nécessaire qu'un agent soit capable de réagir dans un en-

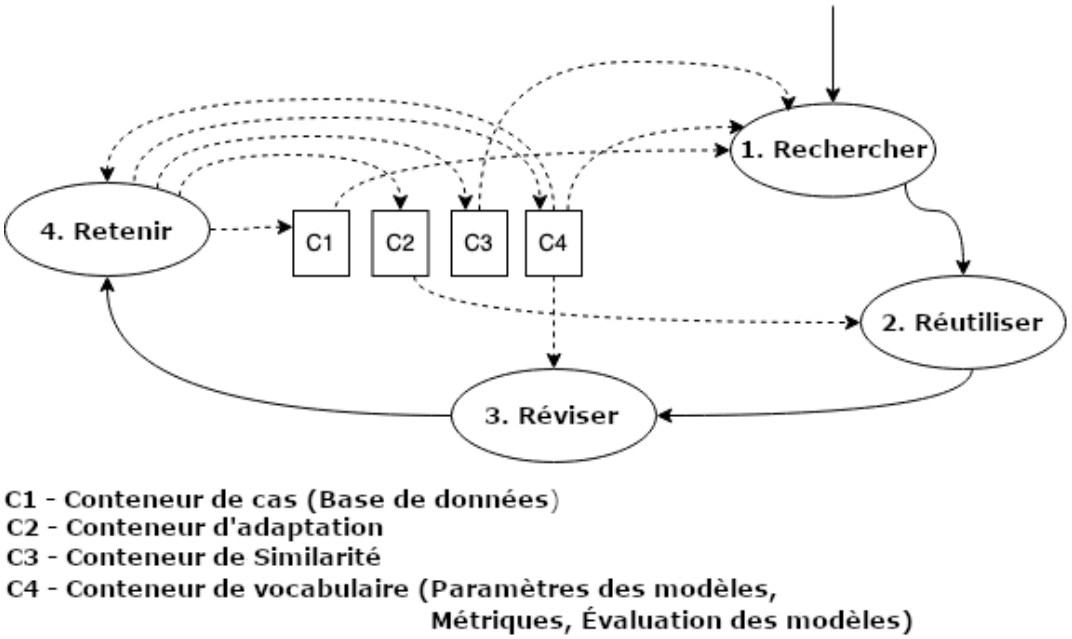

FIGURE 2.5 – Cycle du RàPC, les étapes, les conteneurs et leurs flux de données

vironnement dynamique et inconnu, il doit donc avoir la capacité d'apprendre de ses interactions et ainsi améliorer la qualité de ses réactions et comportements.

Il existe quatre types d'agent en fonction des capacités et des approches :

- Réactif : c'est l'agent qui perçoit constamment l'environnement et agit en fonction de ses objectifs.
- Basé sur les réflexes : c'est l'agent qui considère les options pour atteindre ses objectifs et développe un plan à suivre.
- Hybride : il combine les deux modèles antérieurs en utilisant chacun d'eux en fonction de la situation et de l'objectif.
- Basé sur le comportement : l'agent a à sa disposition un ensemble de modèles de comportement pour réaliser certaines tâches spécifiques. Chaque comportement se déclenche selon des règles prédéfinies ou des conditions d'activation. Le comportement de l'agent peut être modélisé avec différentes stratégies cognitives de pensée ou de raisonnement.

2.2.3/ DIFFÉRENTS ALGORITHMES ET FONCTIONS IMPLÉMENTÉS DANS AI-VT POUR LA PERSONNALISATION ET L'ADAPTATION DES SÉANCES D'ENTRAINEMENT PROPOSÉES

2.2.3.1/ PENSÉE BAYESIENNE

D'après les travaux de certains mathématiciens et neuroscientifiques, toute forme de cognition peut être modélisée dans le cadre de la formule de Bayes (équation 2.2), car elle permet de tester différentes hypothèses et donner plus de crédibilité à celle qui est confirmée par les observations. Étant donné que ce type de modèle cognitif permet l'apprentissage optimal, la prédiction sur les événements passés, l'échantillonnage représentatif

et l'inférence d'information manquante ; il est appliqué dans quelques algorithmes de machine learning et d'intelligence artificielle [Hoang, 2018].

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.2}$$

La formule de Bayes calcule la probabilité a posteriori $P(A|B)$ de la plausibilité de la théorie A compte tenu des données B, et requiert trois termes : (1) une probabilité a priori $P(A)$ de la plausibilité de l'hypothèse A, (2) le terme $P(B|A)$ qui mesure la capacité de l'hypothèse A à expliquer les données observées B et (3) la fonction de partition $P(B)$ qui met en compétition toutes les hypothèses qui ont pu générer les données observées B. À chaque nouvelle évaluation de la formule, la valeur du terme a priori $P(A)$ est actualisée par la valeur du terme a posteriori $P(A|B)$. Ainsi, à chaque évaluation, le degré de plausibilité de chaque hypothèse est ajusté [Hoang, 2018].

Par la suite, nous explicitons quelques algorithmes, intègrés généralement dans l'étape "Rechercher" du RàPC, pour la recherche des cas les plus proches d'un nouveau cas.

2.2.3.2/ MÉTHODE DES K PLUS PROCHES VOISINS (K-NEAREST NEIGHBORHOOD - KNN)

Comme est défini dans [Cunningham and Delany, 2021], la méthode des 'k' plus proches voisins est une méthode pour classifier des données dans des classes spécifiques. Pour faire cela, il est nécessaire de disposer d'une base de données avec des exemples déjà identifiés. Pour classer de nouveaux exemples, la méthode attribue la même classe que celle de leurs voisins les plus proches. Généralement, l'algorithme compare les caractéristiques d'une entité avec plusieurs possibles voisins pour essayer d'obtenir des résultats plus précis. 'k' représente le nombre de voisins, sachant que l'algorithme peut être exécuté à chaque fois avec un nombre différent de voisins.

Étant donné un jeu de données D constitué de $(x_i)_{i\in[1,n]}$ données (où $n = |D|$). Chacune des données est décrite par F caractéristiques qui sont des valeurs numériques normalisées $[0, 1]$, et par une classe de labellisation $y_j \in Y$. Le but est de classifier une donnée inconnue q. Pour chaque $x_i \in D$, il est possible de calculer la distance entre q et x_i selon l'équation 2.3.

$$d(q, x_i) = \sum_{f\in F} w_f \delta(q_f, x_{if}) \tag{2.3}$$

La distance entre q et x_i est la somme pondérée de toutes les distances élémentaires $delta$ calculées pour chaque caractéristique. La fonction de distance δ peut être une métrique générique. Pour des valeurs numériques discrètes il est possible d'utiliser la définition 2.4,

$$\delta(q_f, x_{if}) = \begin{cases} 0 & q_f = x_{if} \\ 1 & q_f \neq x_{if} \end{cases} \tag{2.4}$$

et si les valeurs sont continues l'équation 2.5.

$$\delta(q_f, x_{if}) = |q_f - x_{if}| \tag{2.5}$$

Un poids plus important est généralement attribué aux voisins les plus proches. Le vote pondéré en fonction de la distance est une technique couramment utilisée (équation 2.6).

$$N(a,b) = \begin{cases} 1 & a = b \\ 0 & a \neq b \end{cases} \tag{2.6}$$

Le vote est couramment calculé selon l'équation 2.7.

$$vote(y_i) = \sum_{c=1}^{k} \frac{1}{d(q, x_c)^p} N(y_j, y_c) \tag{2.7}$$

une autre alternative est basée sur le travail de Shepard, équation 2.8.

$$vote(y_i) = \sum_{c=1}^{k} e^{d(q,x_c)} N(y_j, y_c) \tag{2.8}$$

La fonction de distance peut être n'importe quelle mesure d'affinité entre deux objets, mais cette fonction doit répondre à quatre critères (équations 2.9).

$$\begin{aligned} &d(x,y) \geq 0 \\ &d(x,y) = 0, \ \textit{seulement si } x = y \\ &d(x,y) = d(y,x) \\ &d(x,z) \geq d(x,y) + d(y,z) \end{aligned} \tag{2.9}$$

2.2.3.3/ K-MOYENNES

Selon [Sinaga and Yang, 2020], K-Moyennes est un algorithme d'apprentissage utilisé pour partitionner et grouper des données. Considérons $X = \{x_1, ..., x_n\}$ un ensemble de vecteurs dans un espace Euclidien $\mathbb{R}^d$, et $A = \{a_1, ..., a_c\}$ où c est le nombre de groupes. Considérons également $z = [z_{ik}]_{n \times c}$, où z_{ik} est une variable binaire (i.e. $z_{ik} \in \{0, 1\}$) qui indique si une donnée x_i appartient au k-ème groupe, $k = 1, , c$. La fonction bijective k-moyenne est définie selon l'équation 2.10.

$$J(z, A) = \sum_{i=1}^{n} \sum_{k=1}^{c} z_{ik} \parallel x_i - a_k \parallel^2 \tag{2.10}$$

Cet algorithme minimise la fonction objectif des k moyennes $J(z, A)$: à chaque itération, les termes a_k et z_{ik} sont calculés selon les équations 2.11 et 2.12.

$$a_k = \frac{\sum_{i=1}^{n} z_{ik} x_{ij}}{\sum_{i=1}^{n} z_{ik}} \tag{2.11}$$

$$z_{ik} = \begin{cases} 1 & if \parallel x_i - a_k \parallel^2 = min_{1 \leq k \leq c} \parallel x_i - a_k \parallel^2 \\ 0, & otherwise \end{cases} \tag{2.12}$$

2.2.3.4/ MODÈLE DE MÉLANGE GAUSSIEN GMM (*Gaussian Mixture Model*)

Le modèle de mélange gaussien (GMM) est un modèle probabiliste composé de plusieurs modèles gaussiens simples. Ce modèle est décrit dans [Wang et al., 2021]. En considérant une variable d'entrée multidimensionnelle $x = \{x_1, x_2, ..., x_d\}$, GMM multivarié est défini selon dans l'équation 2.13.

$$p(x|\theta) = \sum_{k=1}^{K} \alpha_k g(x|\theta_k) \tag{2.13}$$

Dans cette équation, K est le nombre de modèles gaussiens uniques dans GMM, également appelés composants ; α_k est la probabilité de mélange de la k-ème composante respectant la condition $\sum_{k=1}^{K} \alpha_k = 1$.

θ_k représente la valeur moyenne et la matrice de covariance de chaque modèle gaussien unique : $\theta_k = \{\mu_k, \Sigma_k\}$. Pour un seul modèle gaussien multivarié, nous avons la fonction de densité de probabilité $g(x)$ (équation 2.14).

$$g(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right) \tag{2.14}$$

La tâche principale est d'obtenir les paramètres α_k, θ_k pour tout k défini dans GMM en utilisant un ensemble de données avec N échantillons d'entrainement. Une solution classique pour l'estimation des paramètres requis utilise l'algorithme de maximisation des attentes (Expectation-Maximization EM), qui vise à maximiser la vraisemblance de l'ensemble de données. Il s'agit d'un algorithme itératif durant lequel les paramètres sont continuellement mis à jour jusqu'à ce que la valeur delta log-vraisemblance entre deux itérations soit inférieure à un seuil donné.

2.2.3.5/ FUZZY-C

Fuzzy C-Means Clustering (FCM) est un algorithme de clustering flou non supervisé largement utilisé [Xu et al., 2021]. Le FCM utilise comme mesure de distance la mesure euclidienne. Supposons d'abord que l'ensemble d'échantillons à regrouper est $X = \{x_1, x_2, ..., x_n\}$, où $x_j \in R^d (1 \leq j \leq n)$ dans un espace Euclidien à d dimensions, et c le nombre de clusters. L'équation 2.15 montre la fonction objectif de FCM.

$$J(U, V) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m (x_j - v_i)^T A (x_j - v_i) \tag{2.15}$$

où A est la matrice métrique, m est un nombre quelconque $(m > 1)$ qui dénote le degré de flou, u_{ij} est le degré d'appartenance du j-ème échantillon x_j qui appartient au i-ème cluster, dont le centre est v_i. $U = (u_{ij})$, $V = [v_1, v_2, ..., v_c], 1 \leq i \leq c, 1 \leq j \leq n, 2 \leq c < n$ satisfaisant les conditions de l'équation 2.16.

$$\sum_{i=1}^{c} u_{ij} = 1, u_{ij} \geq 0 \tag{2.16}$$

Pour compléter le modèle, les équations de mise à jour pour le centre du cluster v_i (équation 2.17) et des degrés d'appartenance u_{ij} (équation 2.18) sont définies.

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \tag{2.17}$$

$$u_{ij} = \frac{((x_j - v_i)^T A (x_j - v_i))^{\frac{2}{m-1}}}{\left(\sum_{h=1}^{c} (x_j - v_h)^T A (x_j - v_h)\right)^{\frac{2}{m-1}}} \tag{2.18}$$

Les algorithmes qui se trouvent dans la suite de cette section sont des algorithmes qui font partie de l'intelligence artificielle et sont utilisés dans certains EIAH pour améliorer les recommandations, essayer de corriger et de détecter les faiblesses des apprenants de façon automatique.

2.2.3.6/ BANDIT MANCHOT MAB (*Multi-Armed Bandits*)

Dans [Gupta et al., 2021] MAB est décrit comme un problème qui appartient au domaine de l'apprentissage par renforcement. Celui-ci représente un processus de prise de décision séquentielle dans des instants t de temps, où un utilisateur doit sélectionner une action $k_t \in K$ à réaliser dans un ensemble K fini d'actions possibles et donnant une récompense inconnue de l'utilisateur. L'objectif est de maximiser la récompense cumulée globale dans le temps. La clé pour trouver une solution au problème est de trouver l'équilibre optimal entre l'exploration (exécuter des actions inconnues pour collecter de l'information complémentaire) et l'exploitation (continuer à exécuter les actions déjà connues pour cumuler plus de gains). L'un des algorithmes utilisés pour résoudre ce problème c'est l'échantillonnage de Thompson.

2.2.3.7/ ÉCHANTILLONNAGE DE THOMPSON TS (*Thompson Sampling*)

Comme indiqué dans [Lin, 2022], l'algorithme d'échantillonnage de Thompson est un algorithme de type probabiliste utilisé généralement pour résoudre le problème MAB. Il s'appuie sur un modèle Bayésien dans lequel une distribution de probabilités *Beta* est initialisée. Cette distribution de probabilités est ensuite affinée de manière à optimiser la valeur résultat à estimer. Les valeurs initiales des probabilités de Beta sont définies sur l'intervalle $[0, 1]$. Elle est définie à l'aide de deux paramètres α et β.

L'échantillonnage de Thompson peut être appliqué à la résolution du problème du Bandit Manchot(MAB). Dans ce cas, les actions définies dans le MAB ont chacune une incidence sur la distribution de probabilités Beta de l'échantillonnage de Thompson. Pour chaque action de MAB, les paramètres de Beta sont initialisés à 1. Ces valeurs changent et sont calculées à partir de récompenses obtenues : si au moment d'exécuter une action spécifique le résultat est un succès, alors la valeur du paramètre α de sa distribution Beta augmente mais si le résultat est un échec alors c'est la valeur du paramètre β de sa distribution Beta qui augmente. De cette façon, la distribution pour chacune des actions possibles est ajustée en privilégiant les actions qui génèrent le plus de récompenses.

L'échantillonnage de Thompson est un cas particulier de la loi de Dirichlet comme le montre la figure 2.6. L'équation 6.1 décrit formellement la famille des courbes générées

par la distribution Beta.

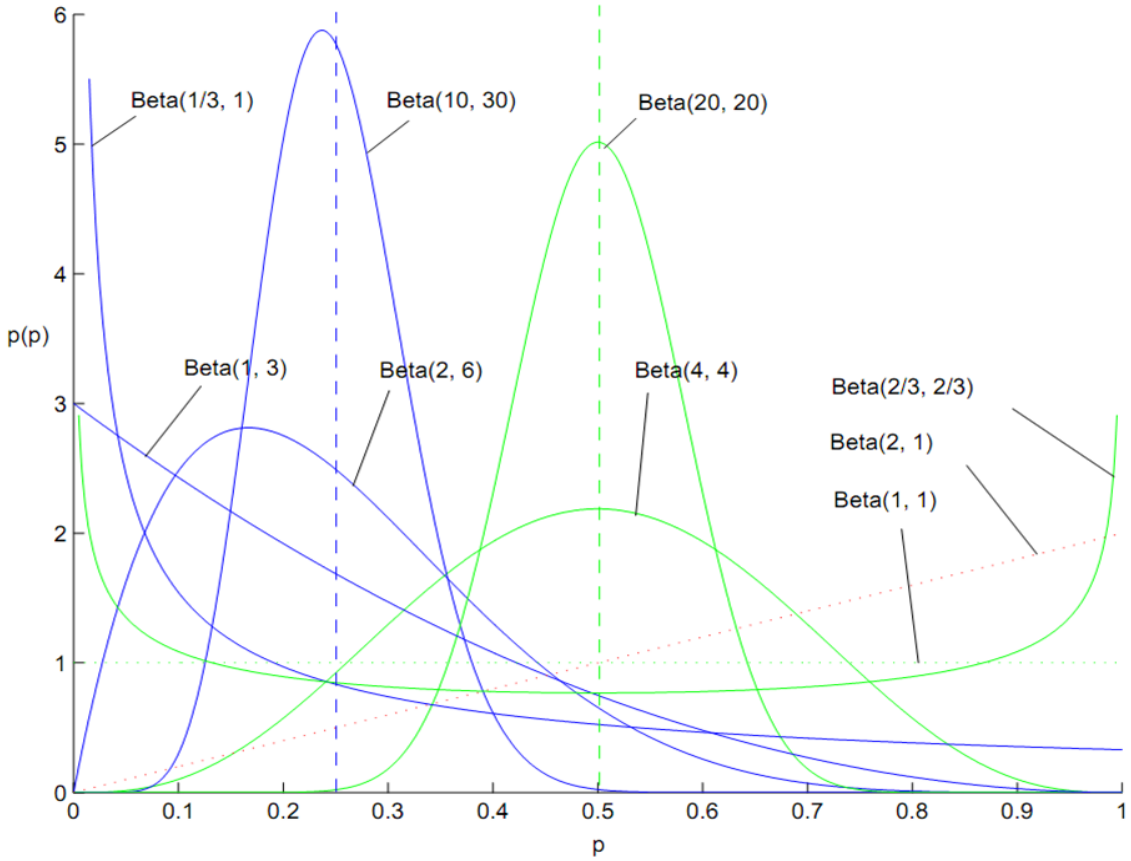

FIGURE 2.6 – Comportement de la distribution Beta avec différentes valeurs de paramètres $alpha$ et $beta$

$$B(x,\alpha,\beta) = \begin{cases} \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}du} & pour x \in [0,1] \\ 0 & sinon \end{cases} \tag{2.19}$$

La section suivante présente un état de l'art plus détaillé des EIAH et des systèmes de RàPC dédiés aux EIAH.

II

ÉTAT DE L'ART

3

Environnements Informatiques d'Apprentissage Humain

Ce chapitre présente les travaux sur les EIAH en lien avec le travail de cette thèse. Les EIAH référencés dans cet état de l'art sont classés selon le thème principal abordé.

3.1/ L'Intelligence Artificielle

L'intelligence artificielle a été appliquée avec succès au domaine de l'éducation dans plusieurs écoles de différents pays et pour l'apprentissage de l'informatique, les langues étrangères et les sciences. [Zhang. and Aslan, 2021] référence un grand nombre de travaux sur la période de 1993-2020. Cet article montre que les besoins et contraintes des EIAH ne sont pas différentes selon l'age, le niveau culturel, ou le niveau éducatif des apprenants. Cet article montre également qu'il est possible d'adapter la stratégie d'apprentissage pour n'importe qui et maximiser le rendement et acquisition des connaissances grâce à différentes techniques d'intelligence artificielle.

[Chiu et al., 2023] présente une révision des travaux d'intelligence artificielle appliquée à l'éducation extraits des bases de données bibliographiques ERIC, WOS et Scopus sur la période 2012-2021. Ce travail est focalisé sur les tendances et les outils employées pour aider le processus d'apprentissage. Une classification des contributions de l'IA est faite, fondée sur 92 travaux scientifiques. Les contributions et l'utilité des outils d'IA implémentés dans les EIAH évalués dans cet article sont évaluées dans les quatre domaines suivants : l'apprentissage, l'enseignement, l'évaluation et l'administration. Pour l'apprentissage, l'IA est généralement utilisée pour attribuer des tâches en fonction des compétences individuelles, fournir des conversations homme-machine, analyser le travail des étudiants pour obtenir des commentaires et accroître l'adaptabilité et l'interactivité dans les environnements numériques. Pour l'enseignement, des stratégies d'enseignement adaptatives peuvent être appliquées pour améliorer la capacité des enseignants à enseigner et soutenir le développement professionnel des enseignants. Dans le domaine de l'évaluation, l'IA peut fournir une notation automatique et prédire les performances des élèves. Enfin, dans le domaine de l'administration, l'IA contribue à améliorer la performance des plateformes de gestion, à soutenir la prise de décision pédagogique et à fournir des services personnalisés. Cet article met également en lumière les lacunes suivantes de l'IA :

— Les ressources recommandées par les plateformes d'apprentissage personnali-

sées sont trop homogènes,
— les données nécessaires pour les modèles d'IA sont très spécifiques,
— le lien entre les technologies et l'enseignement n'est pas bien clair. En effet, la plupart des travaux sont conçus pour un domaine ou un objectif très spécifique difficilement généralisable,
— l'inégalité éducative car les technologies ne motivent pas tous les types d'élèves et parfois il y a des attitudes négatives envers l'IA parfois difficile à maîtriser et à intégrer dans les cours.

Les techniques d'IA peuvent aussi aider à prendre des décisions stratégiques visant des objectifs à longue échéance comme le montre le travail de [Robertson and Watson, 2014]. Les jeux de stratégie présentent en effet plusieurs particularités singulières. Ils continent plusieurs règles et des environnements contraints. Ils nécessitent de mettre en œuvre des actions et réactions en temps réel. Ils intègrent des aléas et des informations cachées. Les techniques principales identifiées et implémentées dans ces jeux de stratégie sont l'apprentissage par renforcement, les modèles bayésiens, la recherche dans une arborescence, le raisonnement à partir de cas, les réseaux de neurones et les algorithmes évolutifs. Il est également important de noter que la combinaison de ces techniques permet dans certains cas d'améliorer le comportement global des algorithmes et d'obtenir de meilleures réponses.

3.2/ SYSTÈMES DE RECOMMANDATION DANS LES EIAH

Des systèmes de recommandation sont régulièrement intégrés aux EIAH. Ils permettent en effet de tenir compte des exigences, des besoins, du profile, des talents, des intérêts et de l'évolution de l'apprenant pour s'adapter et recommander des ressources ou des exercices dans le but d'améliorer l'acquisition et la maîtrise de concepts et des connaissances en général. L'adaptation de ces systèmes peut être de deux types [Muangprathub et al., 2020] :

— l'adaptation de la présentation qui montre aux apprenants les ressources en concordance avec leurs faiblesses et
— l'adaptation de la navigation qui change la structure du cours en fonction du niveau et style d'apprentissage de chaque apprenant.

Ces systèmes montrent des effets positifs pour les apprenants comme le révèle le travail de [Huang et al., 2023] qui utilise l'IA pour personnaliser des recommandations de ressources en vidéo et ainsi aider et motiver les apprenants. L'évolution des résultats entre les tests préliminaires et ceux réalisés après la finalisation du cours des groupes ayant suivi les cours avec et sans système de recommandation démontrent cet effet positif. La figure 3.1 montre l'architecture du système où l'intelligence artificielle est utilisée dans différents étapes du processus avec les données de l'apprenant sous la supervision et contrôle du professeur.

Le travail de [Seznec et al., 2020] propose un modèle générique de recommandation qui peut être utilisé dans les systèmes de recommandation de produits ou les systèmes d'apprentissage. Ce modèle est fondé sur l'algorithme par renforcement UCB (*Upper Confidence Bound*) qui permet de trouver une solution approchée à une variante du problème *Bandit manchot* non stationnaire (MAB, présenté dans la section **??** de ce manuscrit), où les récompenses de chaque action diminuent progressivement après chaque utilisation. Pour valider le modèle, une comparaison avec trois autres algorithmes a été menée en

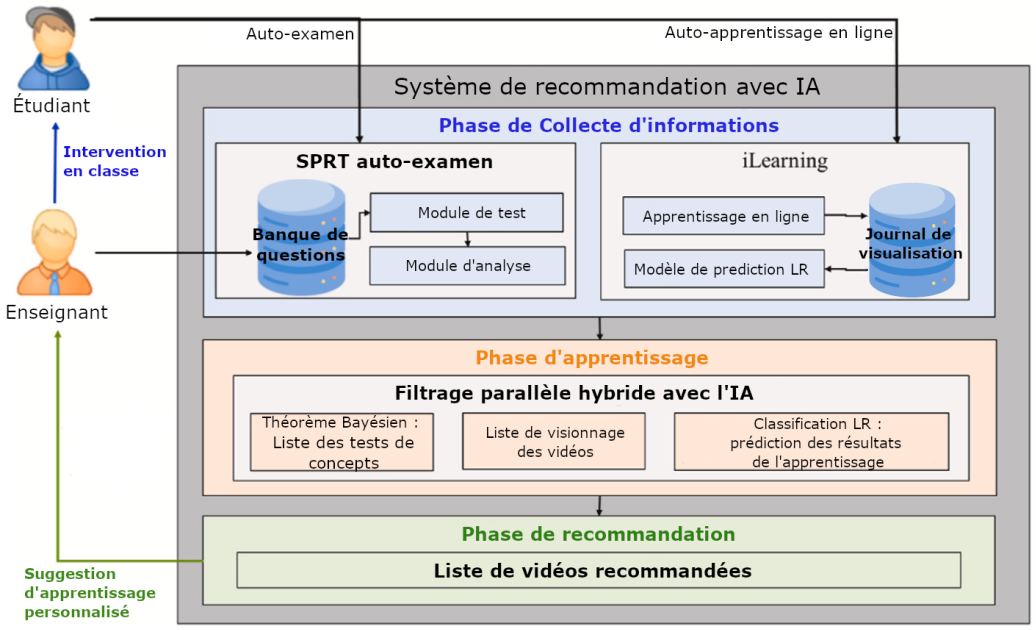

FIGURE 3.1 – Traduction de l'architecture du système de recommandation proposé dans [Huang et al., 2023]

considérant une base de données réelle. La performance du système a été démontrée en se fondant sur la moyenne accumulée du regret et la moyenne de la récompense accumulée.

Aussi [Ingkavara et al., 2022] met en évidence que les technologies et les systèmes de recommandation peuvent s'adapter à différents besoins et aspirations et qu'ils peuvent également favoriser l'apprentissage auto-régulé. Ce type d'apprentissage aide les apprenants à acquérir les habilités pour diminuer les délais de réponse et devenir plus performants. Ce type d'apprentissage permet de définir des objectifs variables dans un environnement structuré, également d'adapter le temps nécessaire à l'acquisition et la maîtrise de connaissances et de compétences. De plus, avec les systèmes de recommandation est possible d'avoir accès à des ressources et ainsi constituer un excellent outil pour le renforcement des connaissances acquises.

L'IA est utilisée pour l'apprentissage adaptatif afin de suggérer des ressources d'étude dans le travail de [Lalitha and Sreeja, 2020]. Le système proposé par ces auteurs intègre un module de personnalisation qui collecte l'information de l'apprenant et ses exigences, un module de classification qui compare l'information avec d'autres profiles en utilisant l'algorithme KNN et un module de recommandation chargé d'identifier les ressources communes entre les profiles afin d'en extraire des informations complémentaires provenant d'Internet. Ce système inclut également un algorithme *Random Forest* pour améliorer le processus d'apprentissage du nouvel apprenant. Les techniques et les stratégies sont très variées mais l'objectif est de s'adapter aux apprenants et à leurs besoins. Dans [Zhao et al., 2023] les données des apprenants sont collectées et classées dans des groupes présentant des caractéristiques similaires. Ensuite, la méthode d'analyse par enveloppement des données (DEA) permet d'identifier les besoins spécifiques de chaque groupe et ainsi proposer un parcours d'apprentissage personnalisé.

La représentation des données des enseignants, des apprenants et l'environnement sont des aspects particulièrement importants à considérer dans l'implémentation de ces systèmes de recommandation. Ces aspects influencent en effet les performances globales des EIAH dans lesquels ils sont intégrés. La proposition de [Su et al., 2022] consiste à stocker les données des apprenants dans deux graphes évolutifs qui contiennent les relations entre les questions et les réponses correctes ainsi que les réponses données par les apprenants. Ces informations et relations permettent de construire un graphe global propre à chaque apprenant et donnant une information sur son état cognitif et ainsi de personnaliser son parcours. Dans [Muangprathub et al., 2020] définit dans son EIAH trois ensembles de concepts :

— les objets (G),
— les attributs (M) et
— les relations entre G et M.

L'originalité de ce travail consiste à analyser ces concepts en via l'approche FCA (*Formal Context Analysis*). Ce type de représentation permet de mettre en relation les ressources d'étude, les questions et les sujets. En conséquence, si un apprenant étudie un sujet S_1 et si une règle relie S_1 au sujet S_4 ou la question $Q_{3,4}$, alors le système peut suggérer l'étude des ressources d'étude associés au sujet S_4 ou aux questions relies par les règles. Par ailleurs, cet algorithme se fonde sur une structure du cours prédéfinie pour recommander un parcours exhaustif d'apprentissage.

[Zhou and Wang, 2021] propose un système de recommandation pour personnaliser des exercices pour l'apprentissage de l'anglais. Le système décrit contient un module principal qui représente les apprenants comme des vecteurs selon le modèle DINA où sont stockes les points de connaissance acquise. Le vecteur de connaissances K est de dimension n ($K = \{k_1, k_2, ..., k_n\}$) où chaque dimension correspond à un point de connaissance. Ainsi, $k_i = 1$ signifie que l'apprenant maîtrise le point de connaissance k_i. A contrario, $k_i = 0$ signifie que l'apprenant doit étudier le point de connaissance k_i car car non acquise. La recommandation des questions proposées par le système se sert par ailleurs d'une librairie de ressources associes à chacune des questions possibles et permettant à l'apprenant de renforcer son apprentissage et d'avancer dans le programme du cours.

Certains travaux de recommandation et de personnalisation considèrent des variables complémentaires aux notes. C'est le cas de l'EIAH proposé par [Ezaldeen et al., 2022], qui lie l'analyse du comportement de l'apprenant à une analyse sémantique de son propre profil. La première étape consiste à collecter les données nécessaires pour créer un profile de l'apprenant. Fort de ce profile complet, des associations sont faites entre l'apprenant et un groupe de catégories d'apprentissages prédéfinies. Les apprentissages sont regroupés selon ses préférences et les données historiques. Puis les concepts associés pour les catégories sont recherchés et permettent ainsi de générer un guide pour obtenir des ressources internet qu'il est possible de recommander. Le système est divisé en quatre niveaux représentés sur la figure 3.2 où chacun des niveaux est chargé d'une tache spécifique. Les résultats d'un niveau sont envoyés au niveau suivant jusqu'à arriver aux recommandations personnalisées pour l'apprenant.

Le tableau 3.1 présente un récapitulatif des articles analysés dans l'état de l'art des EIAH. Il est intéressant de présenter ce tableau, mais je pense qu'il faut que tu commentes ce tableau en prenant le lecteur par la main et en l'aidant à en retenir les informations importantes

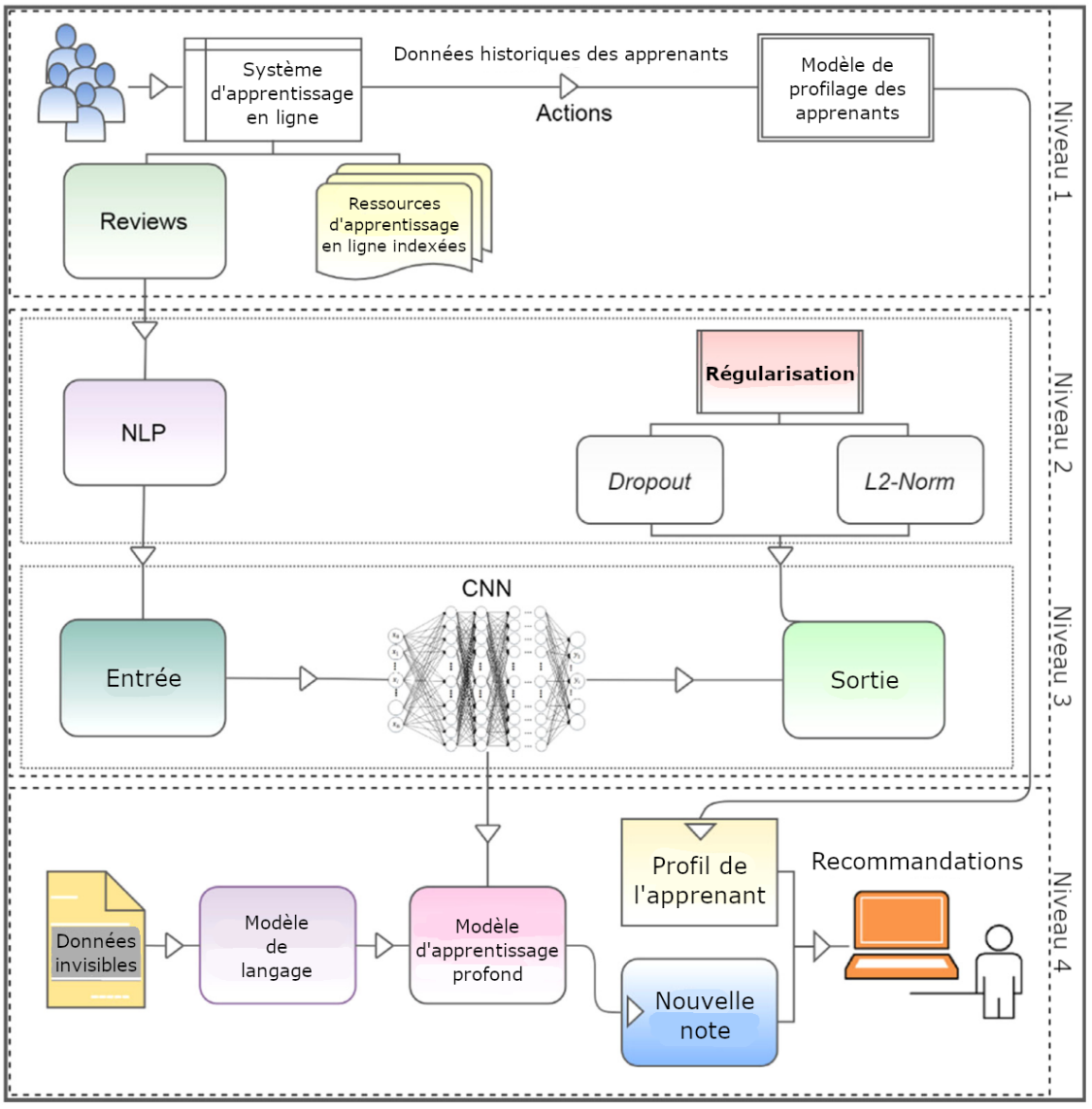

FIGURE 3.2 – Traduction des niveaux du système de recommandation dans [Ezaldeen et al., 2022]

Une limitation générale qui présentent les algorithmes d'IA dans les EIAH est que ce type d'algorithmes ne permettent pas d'oublier l'information avec laquelle ils ont été entrainés, une propriété nécessaire pour les EIAH dans certaines situations où peut se produire un dysfonctionnement du système, la réévaluation d'un apprenant ou la restructuration d'un cours. Par ailleurs, les représentations et algorithmes peuvent changer et évoluer. Aujourd'hui il y a encore beaucoup de potentiel pour développer encore les capacités de calcul et d'adaptation.

Référence	Limites
[Zhang. and Aslan, 2021]	Niveau global d'application de EIAH. Analyse des effets de l'IA
[Chiu et al., 2023]	Aspects non cognitifs en IA. Motivation des apprenants
[Robertson and Watson, 2014]	Peu de test. Pas de standard d'évaluation
[Huang et al., 2023]	Recommandation en fonction de la motivation seulement. N'est pas général pour tous les apprenants
[Seznec et al., 2020]	Le modèle n'a pas été testé avec données d'étudiants. UCB prends beaucoup de temps pour explorer les alternatives
[Ingkavara et al., 2022]	Modèle très complexe. Définition de constantes subjectif et beaucoup de variables.
[Lalitha and Sreeja, 2020]	A besoin de beaucoup de données pour entrainement. Ne marche pas dans le cas 'cold-start', n'est pas totalement automatisé
[Su et al., 2022]	Estimation de la connaissance de façon subjective. Il est nécessaire une étape d'entrainement
[Muangprathub et al., 2020]	Structure des règles complexe. Définition de la connaissance de base complexe
[Zhou and Wang, 2021]	Utilisation d'un filtre collaboratif sans stratification. Utilisation d'une seule métrique de distance
[Ezaldeen et al., 2022]	Il n y a pas de comparaison avec d'autres modèles différents de CNN. Beaucoup de variables à valeurs subjectives

TABLE 3.1 – Tableau de synthèse des articles analysés dans l'état de l'art des EIAH

4

ÉTAT DE L'ART (RAISONNEMENT À PARTIR DE CAS)

4.1/ RAISONNEMENT À PARTIR DE CAS (RÀPC)

Le raisonnement à partir de cas est une approche fondée sur la connaissance. Il s'agit d'une technique d'intelligence artificielle dont l'idée est de résoudre un nouveau problème grâce aux connaissances déjà acquises par le système et en tenant un raisonnement fondé sur l'analogie. Le RàPC est apparu comme une alternative pour améliorer les systèmes experts. Shank et Abelson [Schank and Abelson, 1977] ont initialement mené des travaux sur l'organisation hiérarchique de la mémoire pour imiter le raisonnement humain. Ceux-ci ont servi de fondement aux travaux de Janet Kolodner [Kolodner, 1983] et ont abouti à l'implémentation un système fondé sur ces principes en 1983. Le terme *raisonnement à partir de cas* est utilisé pour la première fois en 1989 par Riesbeck et Shank [C.K. and R.C., 1989].

Comme vu dans le chapitre du contexte, le raisonnement à partir de cas utilise quatre conteneurs de connaissance pour représenter la connaissance complète du système, chaque conteneur stocke l'information associée à une fonction spécifique et tout est fondé sur le carré d'analogie, c'est-à-dire des solutions ayant permis de résoudre un problème ancien sont réutilisées afin de résoudre un problème nouveau similaire. La plupart des systèmes de RàPC utilisent pour son fonctionnement comme base un cycle composé de quatre étapes (retrouver, réutiliser, réviser et retenir).

Les travaux ici cités sont des travaux représentatifs parce que ils ont permis de trouver certaines points d'amélioration du RàPC, ont donné des pistes d'intégration avec d'autres techniques ou ont donné des idées de modification pour obtenir bons résultats en gagnant de la performance.

Plusieurs travaux combinent le RàPC avec les réseaux de neurones (Deep Learning) avec l'objectif d'améliorer les réponses générées et optimiser les performances de chaque algorithme. C'est une idée qui marche dans certains cas, mais qui n'est pas très récente comme par exemple dans [Jung et al., 2009] les auteurs développent un système fondé sur l'hybridation du RàPC avec des réseaux de neurones pour concevoir des produits. Le système se focalise uniquement dans les phases "rechercher" et "réutiliser" du RàPC dans lesquelles sont exécutés les algorithmes implémentés. Le système détermine de façon automatique les valeurs pour les paramètres nécessaires à la conception d'un produit particulier en suivant le cycle traditionnel du RàPC. Avec l'algorithme de

k-moyennes, est extrait un cas représentatif de la base de cas et l'adaptation des solutions des voisins trouvées est faite avec le réseau de neurones RBFN (Radial Basis Function Network). L'évaluation du système est faite en utilisant une base de données contenant 830 tests, les résultats démontrent que les produits conçus s'ajustent avec un grand pourcentage aux normes définies.

Dans [Butdee and Tichkiewitch, 2011] un réseau de neurones classique est implémenté pour définir la géométrie d'une matrice pour l'extrusion de l'aluminium. En effet, actuellement c'est un processus qui se fait manuellement et par essai et erreur. Le RàPC est alors utilisé pour aider à déterminer les valeurs optimales des paramètres du réseau, en utilisant l'information des matrices d'extrusion déjà testées.

Aussi le travail de [Petrovic et al., 2016] où les auteurs proposent un système de raisonnement à partir de cas pour calculer la dose optimale de radiation pour le traitement du cancer. Dans ce domaine particulier de la radiothérapie, administrer une dose nécessite de connaître avec précision le nombre de faisceaux et l'angle de chacun d'eux. L'algorithme proposé tente de trouver la combinaison de valeurs optimales pour ces deux paramètres en utilisant les réseaux de neurones. L'utilisation des réseaux de neurones intervient lors de l'adaptation des cas connus : ils modifient le nombre et les angles des faisceaux. La validation de l'algorithme est évaluée avec une base de 80 cas réels de cancer du cerveau extraits de l'hôpital de Nottingham City. Le nombre de neurones et de couches ont été définis de façon empirique. Les résultats montrent que l'utilisation des cas historiques et la construction des solutions à partir des solutions déjà connues permet une amélioration de 12% concernant la décision du nombre de faisceaux et de 29% concernant la décision liée à leur angle.

Plus récemment se trouvent travaux comme [Wolf et al., 2024] où le RàPC est employée comme technique pour expliquer les résultats générés par un réseau qui classifie les images en fonction de certains attributs et zones dans les images, le résultats permettent conclure que les explications trouvées avec RàPC sont très fidèles aux images testées. Aussi [Parejas-Llanovarced et al., 2024] utilisent le RàPC et ses avantages pour le coupler à un réseau profond (Deep Learing) pour sélectionner la meilleure explication au résultat donné par le classificateur d'images dans ce cas, la base de connaissance sont certaines images avec des étiquettes qui contiennent l'information associée à l'image, la comparaison réalisée avec les algorithmes représentatifs de l'état de l'art suggèrent que la combinaison Deep Learning marche bien dans ce type de problèmes.

Certains travaux ont appliqué le raisonnement à partir de cas à un problème spécifique en modifiant les représentations des cas et des solutions, d'autres ont modifié le cycle conceptuel comme le montre la figure 4.1 avec l'objectif d'obtenir des réponses dynamiques et plus aptes à problèmes complexes. Dans [Grace et al., 2016] est proposé un cycle complémentaire incluant un outil d'apprentissage profond (*Deep Learning*) pour améliorer le résultat du processus du RàPC. Dans [Butdee and Tichkiewitch, 2011], la phase de stockage est modifiée en retenant les cas dont les résultats n'ont pas eu de succès, pour guider le processus dans la fabrications de nouvelles pièces. Enfin, dans [Robertson and Watson, 2014], les auteurs proposent d'ajouter à chaque cas une valeur d'utilité espérée selon chaque action possible. Cet ajout s'accompagne d'une prédiction probabiliste des actions que l'application engendrera en réponse. Cette prédiction probabiliste dépend bien entendu de l'état initial du système avant mise en oeuvre de l'action.

Plusieurs travaux appliquent le RàPC avec succès en proposant des modifications dans chacune des phases ou en combinant différents algorithmes. Certains systèmes de

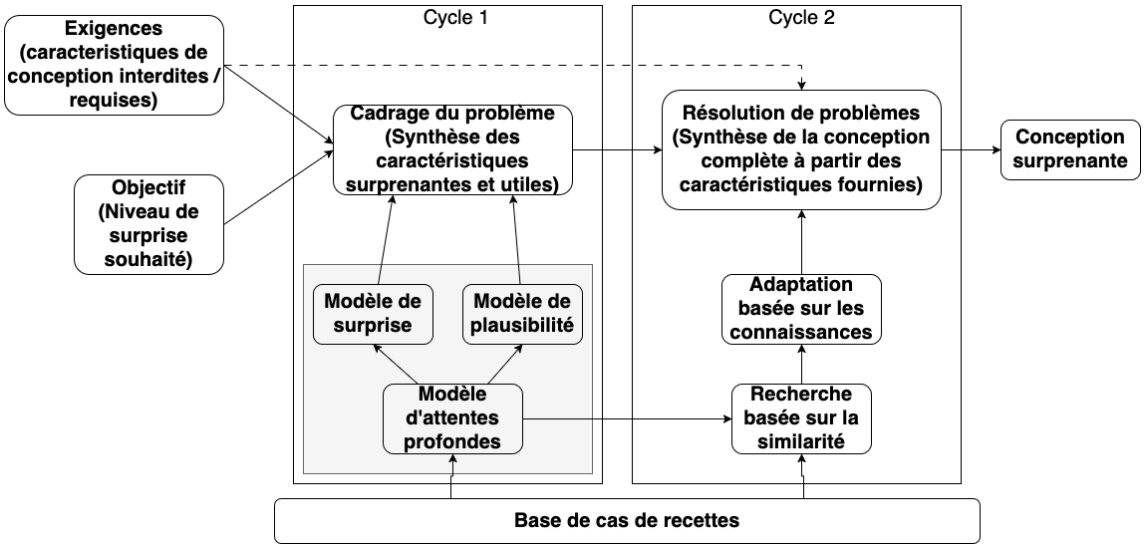

FIGURE 4.1 – Ajout d'un cycle complémentaire avec *Deep Learning* au RàPC (Traduit de [Grace et al., 2016])

RàPC appliqués au domaine de la conception de produits sont remarquables à ce titre. Dans [Roldan Reyes et al., 2015] les auteurs proposent, comme le montre la figure 4.2, un algorithme pour produire le propylène glycol dans un réacteur chimique. Dans ce cas, la phase de réutilisation du RàPC est couplée à la recherche des états qui satisfassent le nouveau problème (*Constraint satisfaction problems CSP*) en utilisant l'information des cas de base déjà résolus. Les solutions trouvées sont évaluées selon le nombre de changements réalisés sur les solutions déjà connues (parcimonie), le nombre de solutions possibles trouvées (précision), l'évaluation de commentaires faits par des experts et la complexité des transformations réalisées.

Dans [Grace et al., 2016], les auteurs ajoutent un nouveau cycle au cycle traditionnel du RàPC. Le premier cycle génère des descriptions abstraites du problème avec un réseau de neurones et des algorithmes génétiques ; le second cycle prend les descriptions abstraites comme des nouveaux cas, cherche les cas similaires et adapte les solutions rencontrées. L'exécution des deux cycles prend en compte certains critères prédéfinis par l'utilisateur. En comparant le même problème avec le cycle traditionnel du RàPC, les auteurs mesurent une amélioration de la qualité des recettes proposées et montrent que celles-ci sont plus en accord avec les critères définis.

[Maher and Grace, 2017] s'intéressent quant à eux à la génération de recettes de cuisine originales. Les auteurs modifient le cycle traditionnel du RàPC et créent deux cycles, combinant l'apprentissage profond et la récupération de cas basée sur la similarité, le premier cycle génère des descriptions abstraites de la conception avec des algorithmes génétiques et un réseau neuronal profond, le second cycle utilise les descriptions abstraites pour récupérer et adapter des objets, cette structure donne lieu à un prototype appelé Q-chef qui génère une recette basée sur la base de données d'ingrédients et les demandes de l'utilisateur. Ce travail ne montre pas de métriques standard génériques mais utilise deux nombres indicatifs (plausibilité et surprise) pour démontrer la génération efficace de nouvelles recettes selon les critères de l'utilisateur en comparant le RàPC à deux cycles avec le RàPC à un cycle, démontrant plus de plausibilité et de surprise dans

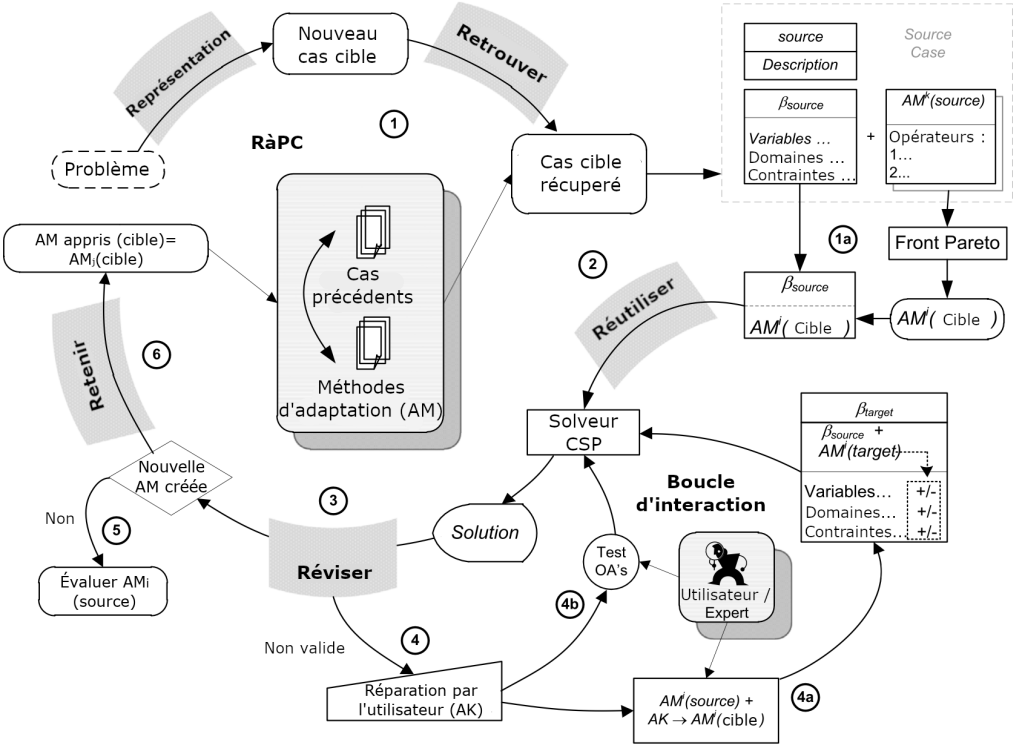

FIGURE 4.2 – Cycle du RàPC modifié. (Traduit de [Roldan Reyes et al., 2015])

les recettes générées.

Par rapport à l'intégration du RàPC avec d'autres algorithmes nous pouvons citer [Uysal and Sonmez, 2023] qui mettent en œuvre un RáPC avec une méthode d'agrégation bootstrap (bagging) pour améliorer la précision du RàPC lorsque il n'y a pas suffisamment de cas et réduire la variance. Le problème qui ils essaient de résoudre est l'estimation des coûts conceptuels dans les décisions de faisabilité d'un projet, lorsque l'on ne dispose pas de suffisamment d'informations sur la conception détaillée et les exigences du projet. Les résultats ont révélé que la performance de prédiction de la méthode RàPC agrégée bootstrap est meilleure que la performance de prédiction de la méthode RàPC traditionnelle.

Un modèle d'ensemble fondé sur le raisonnement à partir de cas est proposé par [Yu and Li, 2023] appliqué à la prédiction financière et au remplissage des données manquantes. Dans ce cas, pour retrouver les plus proches voisins, le modèle utilise trois mesures de distance différentes et une étape de vote pour l'intégration. Le modèle a été testé avec une base de données comportant onze dimensions d'informations financières provenant de 249 entreprises. La comparaison est faite avec deux objectifs. Premièrement, le remplissage des données manquantes avec d'autres algorithmes tels que KNN ou RandomForest, et deuxièmement, la comparaison de la prédiction avec des algorithmes uniques utilisant une métrique de distance spécifique. En effet, les résultats montrent une meilleure performance dans le remplissage des données manquantes et les meilleurs résultats dans la prédiction.

La représentation des cas peut permettre également d'améliorer les résultats d'un système de RàPC. La performance d'un système de RàPC dépend de la quantité d'in-

formations stockées, mais également des algorithmes implémentés. C'est le cas dans [Müller and Bergmann, 2015] où les recettes de cuisine sont codées comme un processus de transformation et mélangent des ingrédients en suivant une suite d'étapes ordonnés. Pour créer des recettes innovantes, une mesure de distance est utilisée entre les ingrédients. Cette mesure permet de trouver une recette en substituant certains ingrédients par d'autres, similaires ou aux qualités et/ou caractéristiques similaires. De la même manière, il est possible de créer des recettes plus proches des exigences des utilisateurs. Les étapes de transformation appelées aussi opérateurs sont stockées et catégorisées grâce à une métrique permettant de les échanger afin d'obtenir une nouvelle recette.

La génération, analyse et correction de texte constitue également un domaine d'application intéressant du RàPC. Pour la réalisation de ces tâches il est parfois nécessaire de transformer le texte en représentation numérique ou de mesurer la proximité sémantique de mots. Le travail de [Ontañón et al., 2015] utilise le RàPC pour générer des histoires en utilisant le texte d'autres histoires. Le système décrit explore les transformations possibles afin que la nouvelle histoire ne soit pas très similaire aux histoires déjà connues mais que elle soit cohérente. Le travail est plus focalisé sur la phase de révision, car les résultats de celle-ci déterminent si l'histoire générée correspond aux critères spécifiés ou si le cycle d'adaptation doit recommencer ; l'adaptation se fait en recherchant les personnages, les contextes, les objets dans la base de cas, et en les unifiant avec des actions ; la plupart des histoires générées sont cohérentes mais elles sont constituées de paragraphes très courts.

Dans [Lepage et al., 2020] les phrases écrites en langue française sont corrigées. Ce travail n'utilise ni la transformation numérique des phrases, ni de connaissances linguistiques, mais retrouve les phrases similaires en utilisant l'algorithme LCS (*Longest Common Subsequence*) et en calculant une mesure de distance avec les phrases correctes et incorrectes de la base de connaissances. Si les phrases similaires sont incorrectes, le système peut proposer une correction en changeant certains mots selon le contexte et recalculer les distances afin de mesurer la cohérence et pertinence de la phrase proposée.

Une étape très importante dans le RàPC est l'adaptation. Dans [Malburg et al., 2024] l'objectif est changer la représentation de la connaissance pour évaluer s'il y a un impact positif en les solutions générées, ce changement permet aussi d'établir règles de adaptation, les résultats permettent de dire que choisir une représentation adéquate et des règles correctes en fonction du problème étudié peut améliorer la qualité des solutions pour résoudre des problèmes complexes.

Étant donne la versatilité du RàPC dans la littérature se trouvent des travaux qui l'utilisent comme un outil de prédiction, car en connaissant l'information historique généralement est possible d'inférer la structure ou le comportement des données et avec le RàPC les résultats sont assez bons.

L'objectif du travail de [Smyth and Cunningham, 2018] est la prédiction du temps de course pour un athlète. L'algorithme KNN est utilisé pour rechercher les cas similaires. Le système interpole un temps final grâce au calcul d'une moyenne pondérée des meilleurs temps des cas similaires retrouvés dans la première étape du RàPC.

Dans [Smyth and Willemsen, 2020], les auteurs essaient de prédire le meilleur temps d'un patineur par analogie avec ceux des patineurs ayant des caractéristiques et une histoire de course similaires. Cependant parfois, calculer une moyenne des temps similaires trouvés ne suffit pas. Certaines caractéristiques liées au contexte, à l'environnement et à

la nature de la course (le type de course, le type de piste, la distance à parcourir, etc.), peuvent en effet influencer de manière importante la performance du patineur. L'algorithme a été testé avec une base de données contenant les informations de 21 courses de 500m, 700m, 1000m, 1500m, 3km, 5km and 10km réalisées entre Septembre 2015 et Janvier 2020.

Un système multi-fonctionnel est décrit dans [Feely et al., 2020]. Celui-ci permet d'obtenir une prédiction du temps de course, de suggérer un plan du rythme de la course et il recommande également un plan d'entraînement pour une course donnée. Les trois fonctionnalités sont implémentées en utilisant le RàPC. Les calculs de la similarité sont fondés sur un historique et des caractéristiques physiques des coureurs. Les plans d'entraînement sont génériques et sont proposés sur les 16 semaines précédant le début du marathon ciblé (selon les auteurs, c'est en effet le temps usuel pour une préparation à ce type d'épreuve). Le système a été évalué avec une base de données constituée de caractéristiques de 21000 coureurs des marathons de Dublin, Londres ou New-York pour la période de 2014 à 2017.

Dans [Yu et al., 2024] est proposé un algorithme en deux stages pour prédire avec précision et robustesse la détresse financière, les deux stages utilisent le RàPC. Le premier stage permet de compléter les valeurs manquantes de la base de données choisie afin de obtenir une meilleure performance fiable et stable, après est exécutée une version du RàPC combinée avec un LVQ (Learning Vector Quantization) algorithme pour faire la prédiction de classification. Les résultats démontrent compétitive solutions pour compléter les valeurs manquantes et classification.

[Sadeghi Moghadam et al., 2024] présentent un nouvel algorithme pour la prévision de la demande de matériel de secours utilisant le RàPC avec la méthode du meilleur-pire (Best-Worst Method - BWM) et les modèles de Markov cachés (Hidden Markov Model - HMM). Le HMM est entrainé avec des données historiques de demande de matériel, une fois l'algorithme détecte une situation de catastrophe, le RàPC cherche les cas similaires pour proposer le matériel nécessaire. D'après les résultats, l'indice d'erreur de prévision est inférieur à 10%, par conséquent, le CBR-BWM-HMM proposé est un algorithme solide et robuste.

Les systèmes de recommandation et le RàPC peuvent aussi être combinés comme dans le système proposé par [Supic, 2018] montre les bénéfices de l'utilisation du RàPC dans les environnements informatiques pour l'apprentissage humain (EIAH). Le modèle proposé suit le cycle traditionnel du RàPC en combinant les modèles d'apprentissages traditionnels et numériques. Les principales contributions sont la représentation des cas et la recommandation des parcours d'apprentissage personnalisés selon les informations issues des autres apprenants. Une base de cas initiaux a été créée pour mesurer l'efficacité du modèle. Celle-ci stocke la recommandation du parcours de 120 apprenants. Des examens sont réalisés avant et après avoir suivi le parcours recommandé par le système permettent de mesurer l'efficacité de la recommandation proposée.

[Obeid et al., 2022]. Le système décrit dans cette étude présente la particularité d'être capable d'analyser des données hétérogènes et multidimensionnelles. Dans ce travail, un parcours de carrière et les universités/collèges est recommandé aux élèves du secondaire en fonction de leurs intérêts. Ce travail montre également une taxonomie des techniques algorithmiques généralement utilisées dans les systèmes de recommandation pour les EIAH (figure 4.3).

En fonction de certains paramètres comme type d'école, nombre d'étudiants, nombre

d'étudiants admis, nombre d'étudiants non-admis, total nombre de classes, etc. [Skittou et al., 2024] développent un modèle hybride entre RàPC et le raisonnement à partir de règles pour recommander des actions de planification de l'éducation en réponse à des cas éducatifs des écoles primaires. Le modèle a été testé des bases de données réelles et a donné de bons résultats avec une grande précision.

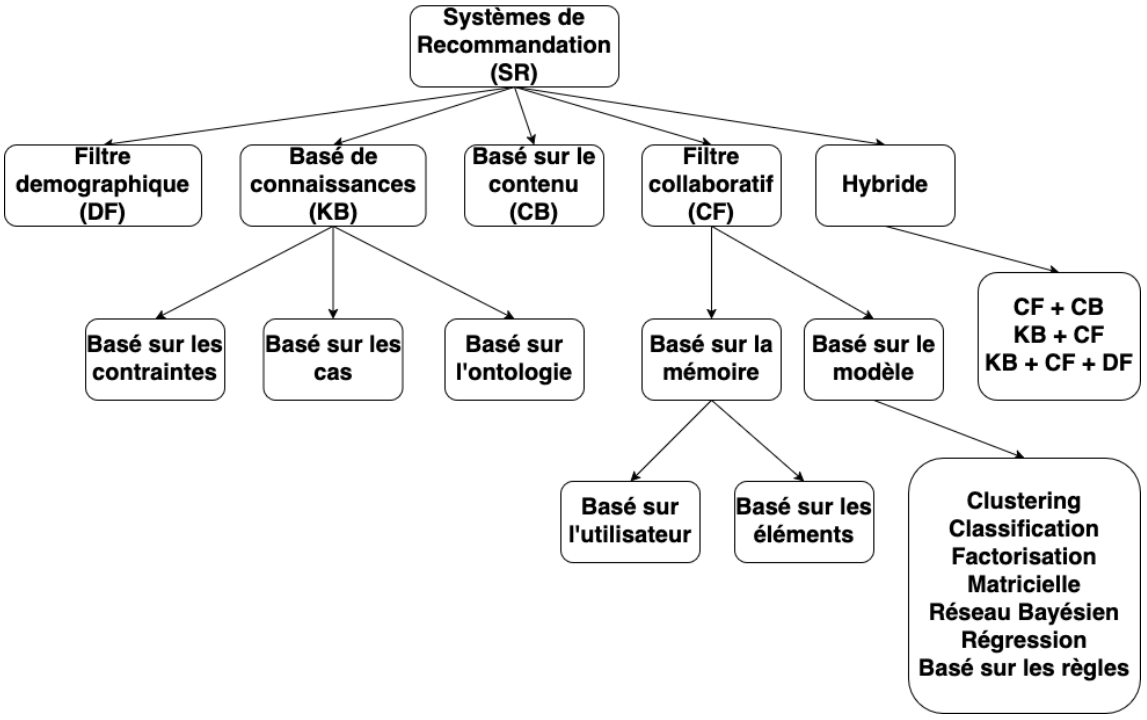

FIGURE 4.3 – Taxonomie des techniques algorithmiques employées pour des modules de recommandation dans les EIAH (Traduit de [Obeid et al., 2022])

Le tableau 4.1 montre un récapitulatif des articles analysés dans cet état de l'art du RàPC, où on montre chacun d'eux en ordre d'apparition dans le chapitre et les limitations trouvées par rapport aux données, flexibilité, généralité de l'algorithme proposé, validité des solutions proposées, automatisation du processus et complexité du modèle proposée. A part les limites identifiés dans le tableau, nous avons trouvé deux problèmes très communs des systèmes de recommandation de façon générale : (1)le *cold-start* qui se produit au début d'une recommandation lorsque il n'y a pas suffisamment de données pour calculer ou inférer une recommandation appropriée [Hu et al., 2025]. Et (2) le *gray-sheep* qui se produit lorsqu'un utilisateur présente un comportement très différent de ceux qui sont stockés dans la base de données. Le système ne peut donc pas générer des recommandations en se basant sur l'information disponible [Alabdulrahman and Viktor, 2021].

Ref	Limites
[Jung et al., 2009]	Le modèle d'adaptation proposé fonctionne seulement avec des cas très proches. L'apprentissage dans le RàCP se limite à stocker les nouveaux cas.
[Butdee and Tichkiewitch, 2011]	Le RàPC n'est pas modifié ou amélioré. La révision dans le RàPC n'est pas automatique
[Petrovic et al., 2016]	Grande quantité de données pour que l'algorithme fonctionne correctement. Recommandation limitée à un problème très spécifique.
[Wolf et al., 2024]	Le modèle n'est actuellement appliqué qu'à la classification d'images à un seul label
[Parejas-Llanovarced et al., 2024]	La base de données de test a été construite avec des évaluations subjectives. Le modèle ne considère pas les incertitudes.
[Roldan Reyes et al., 2015]	Une seule méthode d'adaptation est utilisée. Le modèle n'exploite pas les cas qui présentent une erreur.
[Grace et al., 2016]	Beaucoup de données sont nécessaires pour entrainer le modèle de 'Deep Learning'. Les solutions générées n'ont pas été validées.
[Maher and Grace, 2017]	L'évaluation des solutions générées n'est pas automatique. Les solutions sont produites avec un seul point de vue.
[Uysal and Sonmez, 2023]	La fonction d'unification des solutions est une moyenne des solutions générées avec la division des données et un seul approche
[Yu and Li, 2023]	Une base de données déséquilibrée peut affecter négativement la performance du modèle proposé
[Müller and Bergmann, 2015]	Une seule approche pour adapter les cas dans le RàPC. La révision dans le RàPC n'est pas automatique.
[Ontañón et al., 2015]	Les solutions générées peuvent présenter des incohérences. Il n'y a pas une métrique objective pour mesurer la qualité des réponses.
[Lepage et al., 2020]	Les résultats ne sont pas très bons. Le modèle n'a pas de connaissances linguistiques.
[Malburg et al., 2024]	Les règles d'adaptation définies sont fixes et peuvent être limitées pour certaines situations
[Smyth and Cunningham, 2018]	Les prédictions n'ont pas été testées dans le monde réel. Les données sont très spécifiques et peu variées.
[Smyth and Willemsen, 2020]	Les prédictions sont pour des cas limités. Le modèle est entrainé pour un type d'utilisateur spécifique.
[Feely et al., 2020]	Seulement une technique de sélection de solutions. Les données nécessaires pour le modèle sont complexes.
[Yu et al., 2024]	Il n'y a pas de révision des cas dans la méthode d'imputation ni dans le modèle de classification fondé sur RàPC. Cela peut accroître le biais d'imputation et introduire des échantillons avec bruit
[Sadeghi Moghadam et al., 2024]	Le temps de calcul peut être très grand en fonction de la quantité de données associées, du a l'étape d'entrainement du modèle HMM
[Supic, 2018]	Le modèle a été validé avec peu de données. Les recommandations se basent seulement sur l'information des autres apprenants.
[Obeid et al., 2022]	L'ontologie peut être limitée pour évaluer plusieurs cas inconnus ou imprévus. Il est nécessaire d'avoir une forte connaissance du domaine.
[Skittou et al., 2024]	Les règles de décision établies sont fixes et avec l'hybridation peut avoir de l'incertitude et l'imprécision

TABLE 4.1 – Tableau de synthèse des articles analysés dans l'état de l'art du RàPC

III

CONTRIBUTIONS

5

ARCHITECTURE GLOBALE POUR LE SYSTÈME AI-VT

5.1/ INTRODUCTION

L'architecture logicielle est essentielle pour définir un système entier avant sa mise en œuvre et est importante car elle aide à décrire et à comprendre des systèmes logiciels complexes. L'architecture montre les composants, les couches, les fonctionnalités et les interactions. Ce chapitre présente l'architecture modulaire proposée pour le système AI-VT. Les modules intègrent des paradigmes, des modèles et des algorithmes d'intelligence artificielle pour améliorer le fonctionnement global. Les objectifs de l'architecture proposée sont de rendre le logiciel AI-VT (Artificial Intelligence Virtual Trainer) stable, évolutif, performant, maintenable, facile à modifier, facile à surveiller et testable. L'architecture proposée se compose de quatre couches indépendantes chacune avec une fonctionnalité spécifique : correction automatique, identification, révision et test. Le contenu de ce chapitre est partiellement extrait du travail de Soto *et al.* [Soto-Forero et al., 2024].

Les deux principaux types d'architecture logicielle sont l'architecture monolithique et l'architecture modulaire. Dans l'architecture monolithique, le système logiciel est considéré comme une unité unique avec une seule source de code, une seule base de données et un seul déploiement pour l'ensemble du système ; ce type de système est simple à développer et à tester mais n'est pas adapté à la mise à jour et à l'évolution en raison de sa rigidité. Une architecture modulaire divise le système en modules indépendants qui peuvent communiquer entre eux, chaque module contenant alors tout ce qui est nécessaire pour fonctionner. En fait, de nombreux systèmes logiciels ont été conçus avec une architecture modulaire en raison des multiples avantages qu'elle présente [Auer et al., 2021] [Zuluaga et al., 2022].

Le système AI-VT est un outil pédagogique générique qui vise à accompagner les apprenants dans leur apprentissage en leur proposant des fiches d'exercices appelées sessions. A l'intérieur de chaque session, les capacités attendues sont divisées en compétences, elles-mêmes divisées en sous-compétences. L'apprenant choisit une compétence à travailler et le système génère une session composée d'exercices associés à plusieurs sous-compétences de la compétence choisie. Le système propose une liste d'exercices au début d'une session en utilisant le paradigme du raisonnement par cas avec une base de données de questions.

5.2/ DESCRIPTION DU SYSTÈME AI-VT

Le système AI-VT est un EIAH générique dont la structure globale est présentée dans la figure 5.1. Il existe une base de données de questions, chacune des questions est associée à un contexte, au texte de la question considérée et à un niveau de complexité. Les questions appartiennent à un niveau de sous-compétences et les sous-compétences à un niveau de compétences. Les principaux acteurs du système sont l'enseignant et l'apprenant, l'enseignant a la capacité de configurer l'ensemble du système, le nombre de compétences, les sous-compétences d'une compétence, le nombre de questions, la complexité de chacune d'entre elles, le nombre de niveaux de complexité et le temps par session. Et l'apprenant, qui peut commencer l'entraînement d'une compétence spécifique, accéder à des ressources de soutien complémentaires et répondre aux questions de test dans les sessions proposées par le système. Le tableau 5.1 montre les caractéristiques du système AI-VT selon la définition des 20 dimensions du profil de l'apprenant décrites dans [Jean-Daubias, 2011].

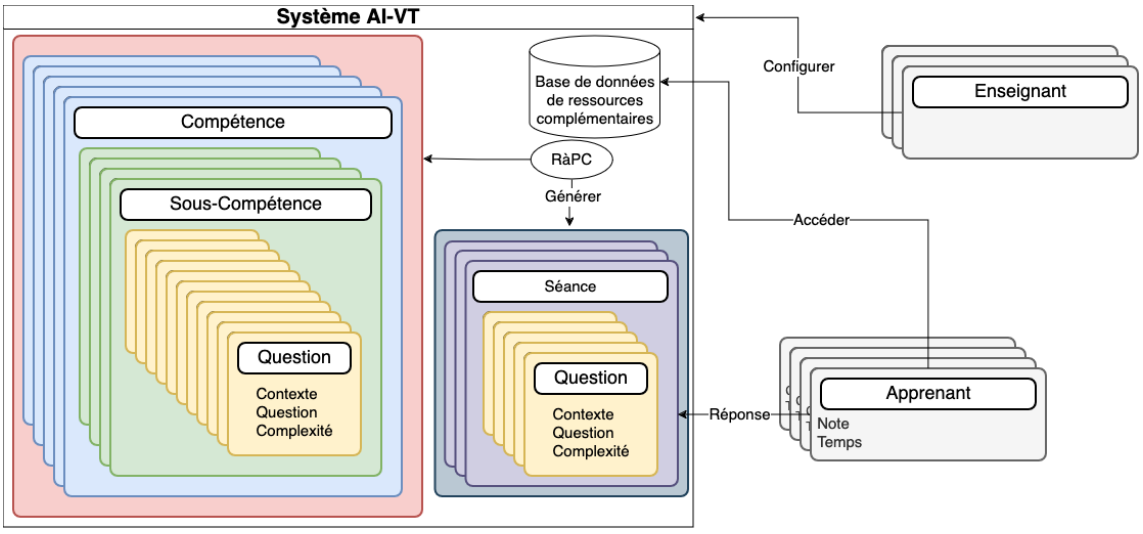

FIGURE 5.1 – Structure du système AI-VT

En s'appuyant sur la philosophie CBR, le système global AI-TV part du principe que certains apprenants ont des performances, des besoins et des capacités d'apprentissage similaires, et qu'il est donc possible de les regrouper et d'améliorer ainsi le processus d'apprentissage général pour tous. L'algorithme dans AI-VT tente de proposer une liste d'exercices en fonction de la compétence ou de la sous-compétence sélectionnée.

L'algorithme d'AI-VT tente de proposer une liste d'exercices en fonction de la compétence ou de la sous-compétence sélectionnée et en tenant compte de l'équilibre entre répétitivité et variété, le nombre d'exercices par session change en fonction du domaine, du niveau et de la compétence de chaque apprenant. La liste d'exercices est générée au début de chaque séance et ne modifie donc pas le cours de la séance en fonction des réponses fournies par l'apprenant, les listes d'exercices sont statiques pendant la séance [Henriet and Greffier, 2018].

Type	Definition	Système AI-VT
Subject	Acteur humain concerné par le profil	Apprentissage seul et en groupe
Collaboration	Le rôle de la collaboration dans les activités du profil	Individuel, Collaboratif
Distance	Le rôle de la distance dans les activités du profil	Presentiel, Distanciel
Discipline	Discipline des informations contenues dans le profil	Generic
Niveau	Le niveau scolaire de la matière concernée par le profil	Generic
Initiateur	L'acteur humain à l'origine de la décision de créer le profil de création	Professeur, Administrateur
Créateur	L'acteur humain ou logiciel qui compose le profil	Professeur, Administrateur
Destinataire	Acteur humain ou logiciel exploitant le profil	Apprenant
Temps	Période du profil	Asynchrone
Évolution	L'évolutivité du profil	Profil évolutif
Type	Le type d'informations contenues dans le profil	Profil de l'apprenant
Nature	La nature des informations contenues dans le profil	Connaissances et compétences
Évaluation	La forme sous laquelle l'information est évaluée	Rating, Taux de maîtrise
Représentation interne	Représentation interne utilisée par le système informatique pour manipuler les profils	Tables
Représentation externe	Représentation utilisée pour stocker le profil	Liste de valeurs
Visualisation	Représentation utilisée pour présenter le profil à ses destinataires	Représentation textuel et graphique standard
Norme	Norme ou standard éducatif	-
Format	Format de stockage du profil	Base de données relationnelle
Plate-forme	plate-forme informatique compatible	Web
Dispositifs	le type de dispositif d'affichage du profil	Ordinateur ou appareils connectés

TABLE 5.1 – Un tableau décrivant les caractéristiques du système AI-VT

5.3/ MODÈLE D'ARCHITECTURE PROPOSÉ

L'idée de l'architecture proposée est que le système préserve sa fonctionnalité d'origine et qu'il peut également utiliser des fonctions complémentaires en activant simplement le module correspondant en envoyant et en recevant les informations nécessaires à son fonctionnement, ce qui étend la fonctionnalité globale d'origine du système et facilite l'in-

tégration avec les modules conçus et même avec de nouveaux modules. La conception modulaire facilite également la maintenance du code, le développement et l'intégration de nouvelles extensions, ainsi que la configuration et l'adaptation du système à différents scénarios, tout en réduisant les risques et les coûts. La modularité permet également d'exécuter les fonctionnalités de chaque module de manière asynchrone, en parallèle ou en mode distribué si nécessaire.

L'architecture se compose de deux éléments principaux : le système central (AI-VT System) et les modules fonctionnels. Le système central gère l'ensemble du processus d'apprentissage ; il génère et démarre les séances ; il stocke les données relatives aux compétences, aux questions, aux ressources, aux apprenants et aux réponses ; il contient les commandes et l'interface générale ; il gère le flux d'informations et active les modules nécessaires. Les modules sont un ensemble de fonctionnalités indépendantes mises en œuvre avec des algorithmes d'intelligence artificielle qui reçoivent et envoient des données depuis le composant central, chaque module fonctionne selon des critères spécifiques liés à son propre objectif. Les modules sont regroupés en couches selon leur fonctionnalité : correction automatique, identification, adaptation, révision et test. L'enseignant et l'apprenant n'utilisent pas les modules directement, les modules sont utilisés par le système pour compléter certaines fonctionnalités.

La couche de correction automatique (LC) correspond aux modules chargés de recevoir les réponses des apprenants et, conformément aux algorithmes et critères définis, d'établir une note cohérente avec une réponse de référence à une question spécifique. Dans cette couche, le module routeur (LC0) est chargé d'identifier le type de correction nécessaire et d'instancier le module approprié pour l'exécution de la tâche spécifique.

La couche d'identification (LD) contient les modules qui identifient les faiblesses ou les variables externes des apprenants lors de l'exécution des exercices proposés par le système ou après l'analyse des résultats. Ces modules aident à personnaliser le processus des apprenants en fonction des résultats obtenus par les analyses.

La couche de révision (LR) comprend les modules qui prennent les données des résultats obtenus dans la couche LC et les résultats de l'analyse de la couche LD pour modifier le parcours de l'apprenant en essayant de renforcer l'apprentissage dans les faiblesses détectées. Ici se trouvent également les modules qui obtiennent des informations de la part des apprenants et tentent de prédire leurs résultats en fonction des différentes compétences et des différents niveaux de complexité.

Pour évaluer les modules dans différents scénarios, il est nécessaire de produire des données selon différents critères et complexités, c'est pourquoi la couche de test (LT) a été définie, dans laquelle se trouvent les modules qui permettent de générer des données synthétiques selon des critères variables, de cette manière il est possible d'obtenir des résultats numériques des modules et d'appliquer des métriques.La couche de test (LT) permet d'évaluer les modules selon différents critères et complexités.

Le schéma complet de l'architecture est illustré à la figure 5.2, où les lignes pleines représentent un flux d'informations bidirectionnel, les lignes pleines avec une flèche représentent le flux unidirectionnel et les lignes en pointillé représentent la dépendance de l'information entre les modules, les dispositifs externes qui peuvent être utilisés par les modules pour exécuter leurs fonctionnalités et les étiquettes qui indiquent quel type d'information le module envoie au système central sont également représentés, ainsi que certains des algorithmes d'intelligence artificielle mis en œuvre dans chaque module et le stade de développement dans lequel chacun d'eux se trouve. Certains dispositifs sont

nécessaires à l'exécution des modules qui requièrent l'obtention de données à partir de sources externes, dans le schéma d'architecture sont représentés le robot NAO, des capteurs, une caméra vidéo et un microphone.

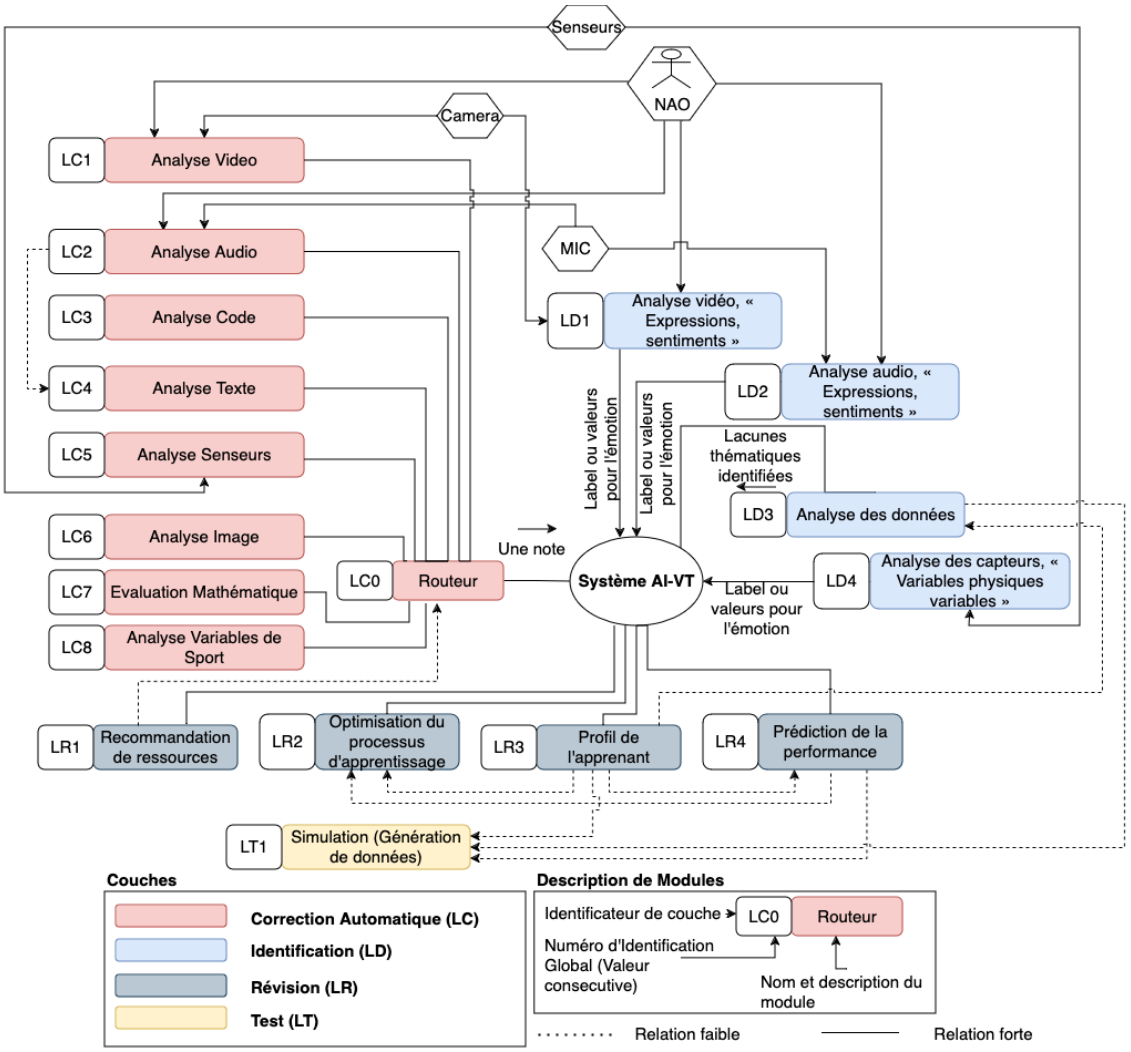

FIGURE 5.2 – Schème de l'architecture proposée

5.3.1/ CORRECTION AUTOMATIQUE

Dans cette couche, les modules ont la capacité de recevoir et d'évaluer différents types de réponses données par les apprenants en fonction du contexte et de la question que le système a proposée. Les modules représentés ont la capacité d'évaluer une réponse donnée par l'apprenant, les modules dans lesquels des progrès ont déjà été réalisés correspondent à l'analyse de vidéo, de texte audio (langage naturel), de code source (Java, Python). D'autres modules permettent également d'analyser des images, des expressions mathématiques, des valeurs générées par des capteurs physiques et des variables définies par des activités sportives.

Le module routeur (LC0) a pour fonction d'identifier le type de réponse et de la rediriger vers le module d'analyse correspondant ; une fois la réponse générée, ce module la

redirige vers le système AI-VT principal.

Le module vidéo (LC1) permet de capturer un flux d'images à partir d'un dispositif externe (caméra vidéo ou robot NAO) et de les analyser pour déterminer si une réponse donnée est correcte, actuellement le module est utilisé pour évaluer la réponse à la question : montrer *n* nombre de doigts. L'algorithme implémenté détecte les doigts qui apparaissent sur la caméra, les compte et détermine s'il s'agit de la bonne réponse à la question donnée.

Le module audio (LC2) analyse une piste audio et tente de convertir son contenu en texte. Une fois le texte généré, il peut être comparé à une réponse attendue à la question posée à l'aide de techniques NLP mises en œuvre dans le module GC5, et un score d'approximation estimé peut être généré. Le module audio (LC2) analyse une piste audio et tente de convertir son contenu en texte.

Le module d'analyse du code (LC3) génère le score après l'exécution de plusieurs étapes, d'abord il détermine s'il y a des faiblesses dans certaines compétences prédéfinies, ensuite il transforme la représentation du code en un vecteur numérique à comparer avec une réponse de référence, le résultat de la pondération entre les faiblesses détectées et le pourcentage de comparaison est le score généré. Le module d'analyse du code (LC3) génère le score après l'exécution de plusieurs étapes.

Le module d'analyse de texte (LC5) transforme essentiellement le texte envoyé par l'apprenant en un vecteur numérique qui peut être comparé au vecteur numérique d'une réponse attendue. Dans ce cas, la réponse donnée ne doit pas nécessairement être exactement la même que la réponse attendue, puisque la représentation vectorielle permet d'établir des similitudes dans l'espace, même si les termes utilisés et la longueur du texte diffèrent.

Les modules LC6, LC7, LC8 et LC9 ont été définis comme des modules capables d'analyser différents types de réponses potentielles, qui ne sont pas encore gérées par le système AI-VT.

5.3.2/ IDENTIFICATION

Les modules de la couche d'identification visent à extraire des informations complémentaires à la note de l'étudiant pour chacune des réponses envoyées au système, principalement pour obtenir des informations sur les expressions et les faiblesses qui peuvent se manifester dans chaque sous-compétence et niveau de complexité, afin d'obtenir une meilleure estimation de l'état de l'apprentissage, pour guider plus précisément le parcours de l'apprenant.

Les modules d'identification LD1, LD2 et LD4 tentent de détecter les comportements, les émotions et les sentiments à l'aide de dispositifs externes tels que la caméra vidéo et le microphone. Dans le cas de l'analyse vidéo, des réseaux neuronaux d'apprentissage profond sont utilisés pour capturer et analyser les images statiques obtenues à partir du flux de la caméra vidéo, les émotions qui peuvent être détectées ont été prédéfinies et le modèle d'IA a été entraîné à les identifier. Le module audio vise à détecter le même type d'émotions, mais à partir de l'analyse des signaux obtenus à partir d'un microphone, il utilise également l'apprentissage profond formé avec des signaux qui présentent les émotions prédéfinies. Le module capteur est conçu de manière plus générique, mais il peut être subdivisé en modules spécifiques en fonction du type de capteur et du signal à

analyser, mais l'idée de la détection est la même avec les émotions prédéfinies.

Le module Analyse des données (LD3) est différent car, en plus d'être générique, il tente d'identifier les faiblesses dans des compétences spécifiques en fonction du type d'évaluation, ce module peut contenir différents modèles entraînés pour chaque type de cas. Pour les exercices linguistiques, les faiblesses doivent être identifiées dans : la conjugaison des verbes, l'utilisation des temps, le vocabulaire, la correspondance des genres de mots, etc. ; si l'exercice est de type programmation : la syntaxe, la déclaration de variables, l'appel de fonctions, la construction de structures, etc. En interne, ce module contient des modèles d'apprentissage profond et des modèles collaboratifs tels que le raisonnement à partir de cas.

5.3.3/ RÉVISION

Dans la couche de révision, les modules utilisent les informations générées par les modules des couches de correction automatique et d'identification ainsi que les informations complémentaires de l'apprenant stockées dans la base de données du système, ces modules valident que la recommandation générée est optimale pour l'apprenant. Toutes les informations récoltées permettent d'établir la meilleure façon de guider l'apprenant vers une meilleure compréhension en surmontant les faiblesses et les lacunes qui ont été identifiées.

Les ressources recommandées par le module LR1 proviennent d'une base de données déjà établie que le module consulte et suggère à l'étudiant en fonction du résultat de la comparaison de l'état d'apprentissage, du niveau et des caractéristiques et spécifications de chacune des ressources.

Le module LR2 a deux variantes, l'une déterministe et l'autre stochastique, les deux variantes étant basées sur les valeurs générées par l'apprenant. Le modèle déterministe utilise un tableau prédéterminé de rangs sur lequel l'apprenant est positionné pour générer la suggestion d'adaptation. Le modèle stochastique utilise des distributions de probabilités dynamiques qui changent en fonction des résultats de l'apprenant à chaque niveau de complexité de la même sous-compétence.

Il y a d'autres modules, où une prédiction est faite avec les mêmes données que celles qui ont été utilisées pour la génération de l'adaptation et de cette façon est déterminée quelle sera la performance de l'apprenant et si effectivement l'adaptation proposée permet d'acquérir les compétences nécessaires et d'améliorer les notes obtenues.

Le module LR3 obtient dynamiquement des structures variables à partir des informations extraites du système AI-VT central. Ce module peut également effectuer des transformations dans la structure, le contenu et la représentation des données des apprenants.

Le module LR4 utilise les informations produites par l'apprenant et les informations collaboratives pour tenter de prédire les performances futures de l'apprenant, en particulier avec la recommandation générée par les modules de la couche adaptative, la prédiction est utilisée pour valider l'adaptation recommandée et l'ajuster si nécessaire.

5.3.4/ TEST

La couche de test est externe au flux du fonctionnement global du système et à l'intégration avec des outils d'intelligence artificielle, mais elle a été développée parce qu'elle permet d'évaluer chacun des modules indépendamment et de générer des données spécifiques pour divers scénarios de test qui peuvent être pris en compte lors de la validation d'un module.

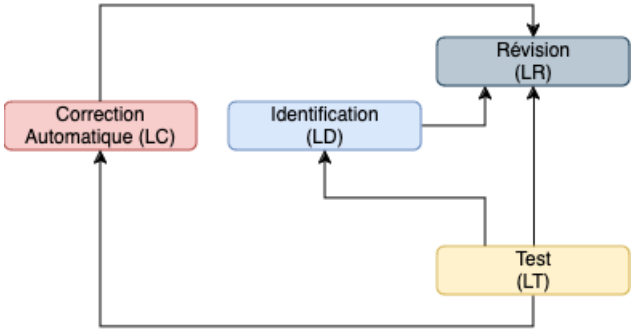

FIGURE 5.3 – Relation entre les couches définies de l'architecture.

Les couches définies de l'architecture peuvent communiquer entre elles, car pour que certains de leurs modules internes fonctionnent, elles ont besoin des informations générées par les modules des autres couches. La figure 5.3 montre les interactions qui peuvent se produire dans le fonctionnement du système. La couche de test (LT) a besoin des informations générées par les modules de toutes les autres couches pour évaluer leurs performances individuelles. La couche d'identification (LD) doit obtenir les données relatives au profil des apprenants ; ces informations sont générées par le module de profil qui se trouve dans la couche de révision (LR), pour certains modules qui ont la capacité de modifier une solution, cette couche doit connaître les résultats obtenus par l'apprenant dans chacun des tests ou exercices proposés par le système, ces résultats sont attribués par la couche de correction automatique (LC). De plus, il est possible d'obtenir une estimation du résultat de la révision proposée avant qu'elle ne soit envoyée à l'apprenant, pour cela il faut invoquer les modules spécifiques de prédiction qui appartiennent à la couche de révision (LR).

Le flux complet d'informations est représenté dans la figure 5.4. La première étape consiste à envoyer le test généré par le système AI-VT à l'apprenant ; à cette étape, le module d'identification correspondant à l'analyse à effectuer est également lancé. Lorsque l'étudiant envoie la réponse à une question, le module d'identification envoie à l'apprenant l'analyse effectuée sur cette même réponse. Grâce à ces informations, le système active les modules de correction automatique pour attribuer une note à la réponse envoyée, en tenant compte également des résultats du module d'identification. À l'étape 6, la note générée est envoyée au système, puis pour effectuer l'adaptation, le système obtient les informations spécifiques de l'apprenant et lance les modules d'adaptation, en envoyant les informations obtenues dans les étapes précédentes. Les algorithmes d'adaptation évaluent les variables et déterminent le parcours optimal pour l'apprenant, mais avant de le renvoyer au système principal, l'étape 9 évalue sa pertinence à l'aide du module de prédiction. Si la prédiction détermine que le chemin suggéré est acceptable, il est envoyé au système principal qui décide, à l'étape 11, de le présenter à l'apprenant comme une alternative au chemin sélectionné à l'origine.

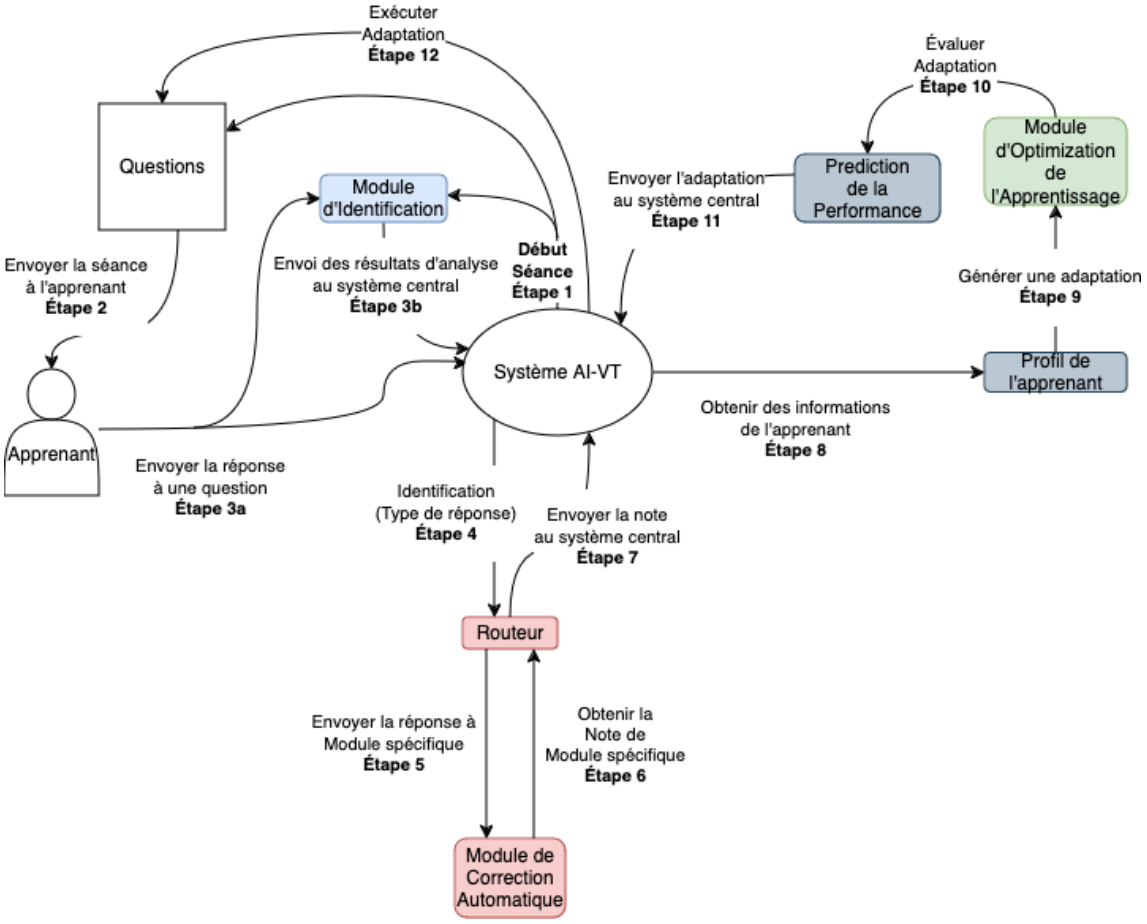

FIGURE 5.4 – Étapes du flux de l'information pour la fonctionnalité de recommandation globale

5.4/ CONCLUSION

L'architecture proposée est basée sur des concepts et des modèles couramment utilisés pour concevoir des systèmes complexes qui utilisent d'une manière ou d'une autre des algorithmes et des outils d'intelligence artificielle. Ce type de conception permet la mise en œuvre d'un système fonctionnel doté d'une capacité d'adaptation, nécessaire à l'exécution de l'une des principales exigences des systèmes d'apprentissage intelligents. En outre, comme l'indiquent les travaux cités en référence, l'architecture modulaire permet une mise en œuvre plus souple et donne au système la possibilité d'évoluer rapidement et même d'ajouter des fonctionnalités complémentaires sans affecter le système et les données qui y sont stockées.

6

SYSTÈME DE RECOMMANDATION DANS AI-VT

6.1/ INTRODUCTION

Ce chapitre explicite l'algorithme de recommandation proposé basé sur les résultats produits par l'apprenant en temps réel. La plupart du contenu est extrait et traduit de l'article Soto *et al.* [**?**].

Ce chapitre présente un modèle d'adaptation automatique en temps réel d'une session prédéterminée à l'intérieur du système AI-VT. Dans cette adaptation le processus fait partie d'un modèle global de raisonnement à partir de cas. Le modèle proposé est stochastique et a été testé avec trois scénarios différents. Les résultats montrent l'adaptation dynamique du modèle proposé, les adaptations obtenues aidant le système à évoluer plus rapidement et identifier les faiblesses des apprenants dans les différents niveaux de complexité ainsi que la génération de recommandations pertinentes dans des cas spécifiques pour chaque capacité d'apprenant.

Le module mis en œuvre pour AI-VT est classé dans la catégorie des systèmes de recommandation. Les systèmes de recommandation dans les environnements d'apprentissage prennent en compte les exigences, les besoins, le profil, les talents, les intérêts et l'évolution de l'apprenant pour adapter et recommander des ressources ou des exercices dans le but d'améliorer l'acquisition et la maîtrise des concepts et des connaissances en général. L'adaptation de ces systèmes peut être de deux types, l'adaptation de la présentation qui montre aux apprenants des ressources d'étude en fonction de leurs faiblesses et l'adaptation de la navigation qui change la structure du cours en fonction du niveau et du style d'apprentissage de chaque apprenant [Muangprathub et al., 2020].

Les techniques de recommandation sont utiles dans les EIAH car elles peuvent détecter les changements et évoluer vers un état optimal, comme l'algorithme d'échantillonnage de Thompson (TS), qui est un algorithme de type probabiliste appartenant à la catégorie des algorithmes d'apprentissage par renforcement, où l'algorithme choisit au temps t une action a à partir d'un ensemble A, obtient une récompense pour l'action a et, en fonction de la valeur de la récompense, ajuste sa stratégie de décision pour choisir au temps $t+1$ une autre action a, dans le but de maximiser la récompense. Il est fondé sur le principe Bayésien, où il y a une distribution de probabilité a priori et avec les données obtenues une distribution de probabilité a posteriori est générée qui vise à maximiser l'estimation de la valeur attendue. Pour la variante de Bernoulli, où la récompense n'a que deux valeurs

possibles 0 et 1 ou succès et échec, la distribution de base utilisée est la distribution Beta qui est définie sur [0, 1] et paramétrée par deux valeurs α et β [Lin, 2022].

6.2/ MODÈLE PROPOSÉ

Le modèle proposé, en tant que système de recommandation, prend en compte les notes antérieures des apprenants pour estimer leurs connaissances et leur maîtrise des différentes compétences, sous-compétences et niveaux de complexité au sein du système AI-VT, puis adapte les sessions pour maximiser l'acquisition des connaissances et la maîtrise des différents domaines contenus dans la même compétence définie. Le modèle est conçu comme une modification de l'algorithme d'échantillonnage de Thompson avec l'intégration de l'échantillonnage stratifié pour obtenir l'adaptation.

On utilise la famille Beta de distributions de probabilité pour définir dynamiquement le nouveau niveau de complexité (équation 6.1) inspiré de l'algorithme d'échantillonnage de Thompson. Cette version du modèle permet de recommander des niveaux de complexité non contigus, mais la priorité est de recommander les niveaux dans lesquels des défauts ont été détectés. La paramétrisation initiale de toutes les distributions de probabilité peut forcer le modèle à recommander des niveaux de complexité contigus plus élémentaires.

$$B(x,\alpha,\beta) = \begin{cases} \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}du} & for\ x \in [0,1] \\ 0 & otherwise \end{cases} \tag{6.1}$$

Les variables qui font partie du modèle sont spécifiées dans le tableau 6.1.

Dans ce cas, il est nécessaire d'utiliser la variable de seuil de grade g_t pour déterminer la variabilité de la distribution de probabilité pour chaque niveau de complexité. Les équations 6.2, 6.3 et 6.4 montrent les règles de mise à jour corrélées, ces règles modifient les valeurs par récompense inverse. Chaque niveau de complexité est associé à une distribution de probabilité Beta avec des valeurs initiales prédéfinies pour les paramètres α et β.

$$ng_c = g_c \tag{6.2}$$

$$ng_c \geq g_t \rightarrow \begin{cases} \beta_c = \beta_c + \Delta_s \\ \beta_{c-1} = \beta_{c-1} + \frac{\Delta_s}{2} \\ \alpha_{c+1} = \alpha_{c+1} + \frac{\Delta_s}{2} \end{cases} \tag{6.3}$$

$$ng_c < g_t \rightarrow \begin{cases} \alpha_c = \alpha_c + \Delta_s \\ \alpha_{c-1} = \alpha_{c-1} + \frac{\Delta_s}{2} \\ \beta_{c+1} = \beta_{c+1} + \frac{\Delta_s}{2} \end{cases} \tag{6.4}$$

Le nouveau niveau de complexité est l'indice de la valeur aléatoire maximale (générée à partir de la distribution Beta de chaque niveau de complexité, équation 6.5) pour tous les niveaux de complexité (équation 6.6).

ID	Description	Domain
c_n	Niveaux de complexité	$\mathbb{N} \mid c_n > 0$
g_m	Valeur maximale dans l'échelle des notes	$\mathbb{N} \mid g_m > 0$
g_t	Seuil de notation	$(0, g_m) \in \mathbb{R}$
s	Nombre de parcours définis	$\mathbb{N} \mid s > 0$
s_c	Parcours courant fixe défini	$[1, s] \in \mathbb{N}$
Δs	Pas pour les paramètres de la distribution bêta dans le parcours s	$(0, 1) \in \mathbb{R}$
t_m	Valeur maximale du temps de réponse	$\mathbb{R} \mid t_m > 0$
g_c	Note de l'apprenant à une question de complexité c	$[0, g_m] \in \mathbb{R}$
ng_c	Grade de l'apprenant avec pénalisation du temps	$[0, g_m] \in \mathbb{R}$
t_c	Le temps de réponse à une question de complexité c	$[0, t_m] \in \mathbb{R}$
ncl	Nouveau niveau de complexité calculé	$\mathbb{N}$
α_c	Valeur de α dans la complexité c	$\mathbb{R} \mid \alpha_c > 0$
β_c	Valeur de β dans la complexité c	$\mathbb{R} \mid \beta_c > 0$
$\Delta\beta$	Pas initial du paramètre bêta	$\mathbb{N} \mid \Delta\beta > 0$
λ	Poids de la pénalisation temporelle	$(0, 1) \in \mathbb{R}$
G_c	Ensemble de d notes dans le niveau de complexité c	$\mathbb{R}^d, d \in \mathbb{N} \mid d > 0$
x_c	Notes moyennes normalisées	$[0, 1] \in \mathbb{R}$
n_c	Nombre total de questions dans une session	$\mathbb{N} \mid n_c > 0$
ny_c	Nombre de questions dans le niveau de complexité c	$\mathbb{N} \mid 0 < ny_c \leq n_c$
y_c	Proportion de questions dans le niveau de complexité c	$[0, 1] \in \mathbb{R}$
r	Valeur totale de la métrique définie pour l'adaptabilité	$[0, c_n] \in \mathbb{R}$
sc	Valeur totale de la métrique de similarité cosinus	$[-1, 1] \in \mathbb{R}$

TABLE 6.1 – Variables et paramètres du modèle proposé

$$\theta_c = Beta(\alpha_c, \beta_c) \tag{6.5}$$

$$ncl = max_x(\mathbb{E}[\theta_x]), 0 <= x <= c_n \tag{6.6}$$

La note des apprenants peut considérer aussi le temps de réponse comme une pénalité, et dans ce cas-là la note est calcule comme la équation 6.7.

$$ng_c = g_c - \left(g_c * \lambda * \frac{t_c}{t_m}\right) \tag{6.7}$$

Le détail des pas d'exécution du modèle proposé sont dans l'algorithme 1.

6.3/ RÉSULTATS

Le comportement du modèle a été testé avec un jeu de données généré, ce jeu de données contient les notes et les temps de réponse de 1000 apprenants pour 5 niveaux de complexité différents, la description des données est indiquée dans le Tableau 6.2. Les notes des apprenants sont générées avec la distribution logit-normale de probabilité, car c'est expérimentalement le meilleur modèle de représentation [**?**].

Algorithm 1 Stochastic Recommendation Model

Initialize the a-priori distributions of probability
for each questions q **do**
  With i as actual complexity level c
  Calculate ng_i ▷ eq 6.2 or eq 6.7
  Update parameters α_i and β_i ▷ eq 6.3 and eq 6.4
  Get random values θ_c with Beta distribution ▷ $\forall c$, eq 6.5
  Get ncl ▷ eq 6.6
end for

L'ensemble de données généré est une simulation des notes des apprenants pour les réponses à quinze questions à chacun des cinq niveaux de complexité. L'ensemble de données simule, via la distribution de probabilité logit-normale, une faiblesse dans chaque niveau de complexité pour 70% des apprenants dans les dix premières questions. La difficulté de la complexité est également simulée en réduisant le score moyen et en augmentant la variance. La figure 6.1 montre la distribution de l'ensemble de données des notes de 1000 apprenants par niveau de complexité.

ID	Description	Domain
q_c	Niveau de complexité de une question q	$[0, c_n] \in \mathbb{N}$
$q_{g,c}$	Note obtenue g pour la question q avec complexité c	$[0, g_m] \in \mathbb{R}$
$q_{t,c}$	Temps employé t pour une question q avec complexité c	$[0, t_m] \in \mathbb{R}$

TABLE 6.2 – Description des variables utilisées dans la base de données evaluée

Toutes les valeurs des paramètres pour tester le modèle sont dans le tableau 6.3.

ID	c_n	g_m	t_m	s	s_c	λ	g_t	$\alpha_{x,1}$	$\alpha_{x,y}$	$\beta_{x,1}$	$\Delta\beta_{x,y}$	Δ_1	Δ_2	Δ_3
Valeur	5	10	120	3	2	0.25	6	2	1	1	1	0.3	0.5	0.7

TABLE 6.3 – Valeurs des paramètres pour les scénarios evalués

Les résultats de la première comparaison sans données historiques (démarrage à froid) entre le modèle proposé, un système de recommandation déterministe et le système original (CBR) sont présentés dans la figure 6.2, où apparaissent différents nombres et échelles de transitions, le système original ne présente pas de transitions, tous les apprenants sont évalués au niveau de complexité 0, les notes obtenues pendant la session ne sont pas prises en compte. Le système avec des modèles de recommandation tente d'adapter le niveau de complexité en fonction des notes obtenues. Le modèle déterministe génère quatre grandes transitions avec un grand nombre d'apprenants dans les questions 5, 6, 8 et 12, toutes entre des niveaux de complexité contigus, la tendance est à la baisse pour les niveaux 0, 1 et 2 après la huitième question et à la hausse pour les niveaux 1 et 3. Le modèle proposé (stochastique), commence par proposer tous les niveaux de complexité possibles mais se concentre sur le niveau 0, les transitions sont constantes mais pour un petit nombre d'apprenants, la tendance après la dixième question est à la baisse pour les niveaux 0 et 4 et à la hausse pour les niveaux 1, 2 et 3. La tendance est à la baisse pour les niveaux 0 et 4 et à la hausse pour les niveaux 1, 2 et 3. La tendance est à la hausse pour les niveaux 1, 2 et 3.

Après la génération de la première session, le système peut continuer avec la liste sui-

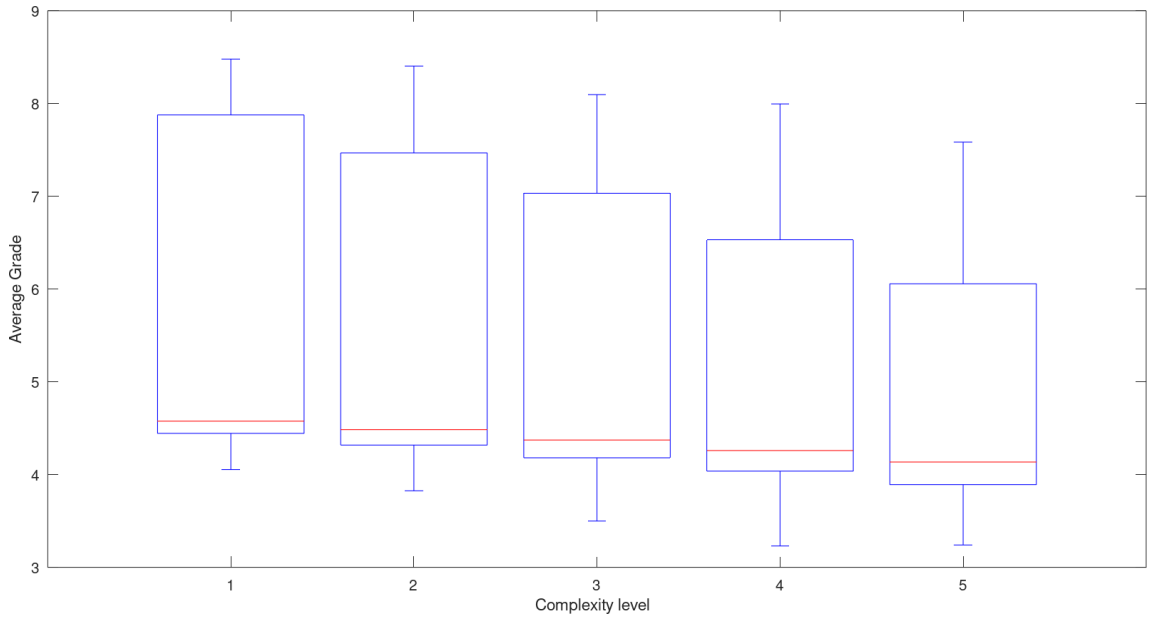

FIGURE 6.1 – Boîte à moustaches pour la base de données générée

vante d'exercices, dans ce cas les trois modèles ont été initialisés avec les mêmes données, et des valeurs égales pour tous les apprenants. La figure 6.3 permet de voir la première transition du système original, cette transition montre que le système agit uniquement avec les notes obtenues dans le passé et les transitions sont très lentes, même si les notes sont différentes au cours de la session, tous les apprenants doivent suivre le même chemin. Cependant, les modèles de recommandation changent, le modèle déterministe présente trois transitions dans les questions 3, 5 et 12. Les tendances sont statiques pour le niveau 3, variables pour le niveau 2 et fortement descendantes pour le niveau 0. Le modèle stochastique continue avec des transitions douces mais essaie toujours de préférer le niveau le plus faible, dans ce cas le modèle a identifié le niveau de complexité 1. Ici, les niveaux 0 et 1 sont descendants, le niveau 2 est statique et les niveaux 3 et 4 sont ascendants.

Enfin, les données d'initialisation considèrent comme évalués deux niveaux de complexité 0 et 1, alors naturellement le système doit commencer avec le niveau 1 ou 2. Comme le système original est très lent à passer d'un niveau à l'autre, ce système commence par le niveau de complexité 1, comme le montre la figure 6.4, comme les deux autres comparaisons, les changements dans ce système ne sont pas progressifs, mais directs pour tous. Dans ce cas, le modèle de recommandation déterministe adopte la même stratégie et propose un changement direct pour tous les apprenants autour de la cinquième question. Le modèle stochastique continue avec des changements faibles mais constants, mais avec une préférence pour le niveau 2, la tendance est très stable sauf pour 1 (à la hausse) et 2 (à la baisse) niveaux.

Pour comparer numériquement le système original, le modèle déterministe et le modèle de recommandation proposé, un ensemble d'équations a été défini (équation 6.8 et équation 6.9) qui décrit le système de recommandation idéal si l'objectif de l'apprenant est l'apprentissage standard, la métrique calcule une valeur pour chaque niveau de complexité en fonction de la moyenne des notes et du nombre de questions recommandées dans ce niveau de complexité. L'objectif de cette métrique est d'attribuer un score élevé

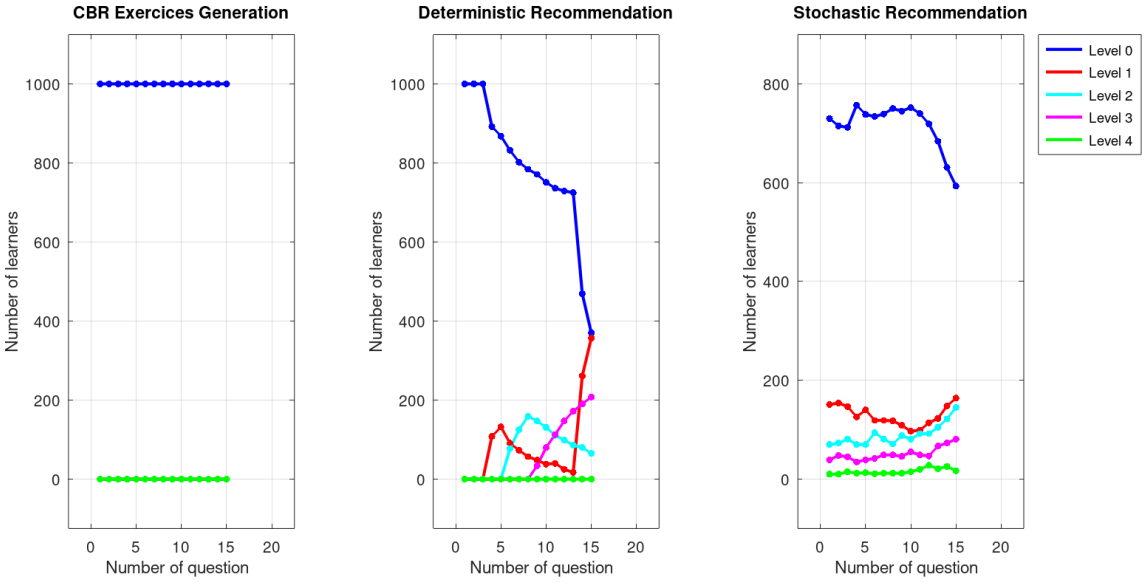

FIGURE 6.2 – Résultats pour le premier test

aux systèmes de recommandation qui proposent plus d'exercices au niveau de complexité où l'apprenant a obtenu une note moyenne plus basse, dans l'idée de renforcer les connaissances à ce niveau de complexité, de même s'ils proposent moins d'exercices aux niveaux de complexité où la note moyenne est élevée, puisqu'il est supposé que l'étudiant a déjà acquis des connaissances suffisantes à ces niveaux de complexité. Les scores faibles sont attribués aux systèmes qui recommandent peu d'exercices à des niveaux de complexité dont les notes moyennes sont faibles et, inversement, s'ils proposent beaucoup d'exercices à des niveaux de complexité dont les notes moyennes sont élevées.

$$rp_c(x) = e^{-2(x_{0,c}+x_{1,c}-1)^2}; \{x \in \mathbb{R}^2 | 0 <= x <= 1\} \tag{6.8}$$

$$r = \sum_{c=0}^{c_n-1} rp_c \tag{6.9}$$

Les propriétés de la métrique sont :

- $\{\forall x \in \mathbb{R}^2 | 0 <= x <= 1\}, rp_c(x) > 0$
- $max(rp_c(x)) = 1;\ if\ x_{0,c} + x_{1,c} = 1$
- $min(rp_c(x)) = 0.1353;\ if\ \left(\sum_{i=1}^{2} x_{i,c} = 0\ \vee\ \sum_{i=1}^{2} x_{i,c} = 2\right)$

Dans l'équation 6.8, $x_{0,c}$ est la moyenne normalisée des notes dans le niveau de complexité c (équation 6.10), et $x_{1,c}$ est le nombre normalisé de questions répondues dans le niveau de complexité c (équation 6.11).

$$x_{0,c} = \frac{< g_c >_{G_c}}{g_m} \tag{6.10}$$

$$x_{1,c} = \frac{ny_c}{n_c} \tag{6.11}$$

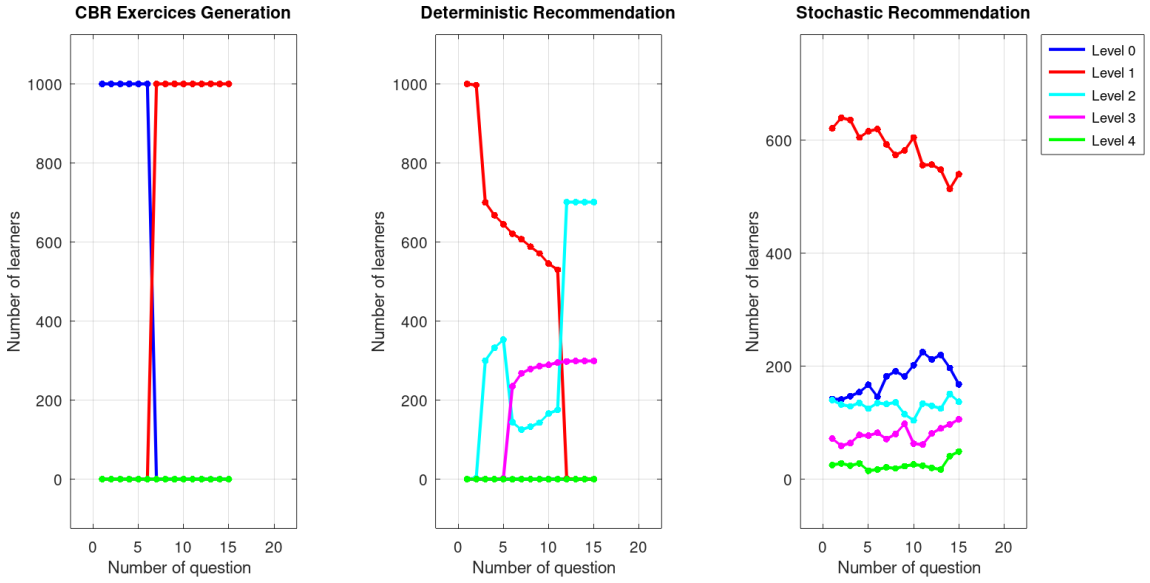

FIGURE 6.3 – Résultats pour le deuxième Test

La figure 6.5 montre l'équation globale pour la métrique rp dans le domaine de deux variables $x_{0,c}$ et $x_{1,c}$. La valeur maximale de r dans un niveau de complexité spécifique est de 1, la valeur maximale globale pour les scénarios testés est de 5. Un bon système de recommandation devrait donc avoir une valeur r élevée.

Les résultats des calculs de la métrique établie pour le système original et les deux modèles dans les trois scénarios définis sont présentés dans le tableau 6.4.

	c_0	c_1	c_2	c_3	c_4	Total (r)	Total (%)
Test 1							
CBR	0.5388	-	-	-	-	0.5388	10.776
DM	0.8821	0.7282	**0.9072**	**0.8759**	-	3.3934	67.868
SM	**0.9463**	**0.8790**	0.7782	0.7108	0.6482	**3.9625**	79.25
Test 2							
CBR	0.9445	**0.9991**	-	-	-	1.9436	38.872
DM	-	0.9443	**0.8208**	**0.9623**	-	2.7274	54.548
SM	**0.9688**	0.9861	0.8067	0.7161	0.6214	**4.0991**	81.982
Test3							
CBR	-	0.8559	0.7377	-	-	1.5936	31.872
DM	-	-	0.5538	**0.7980**	-	1.3518	27.036
SM	0.9089	**0.9072**	**0.9339**	0.7382	0.6544	**4.1426**	82.852

TABLE 6.4 – Résultats de la métrique $rp_c(x)$ (CBR - Système sans modèle de recommandation, DM - Modèle deterministique, SM - Modèle stochastique)

Une métrique pour l'apprentissage en douceur est définie dans l'équation 6.12 et l'équation 6.13, avec cette métrique un score élevé est attribué aux systèmes qui proposent plus d'exercices dans un niveau de complexité où les notes moyennes sont d'environ 0,4 et qui sont plus flexibles avec des notes moyennes plus basses, également si le nombre d'exercices proposés est faible pour des notes moyennes élevées. Les scores faibles sont attribués aux systèmes qui recommandent un nombre élevé de questions dans un niveau

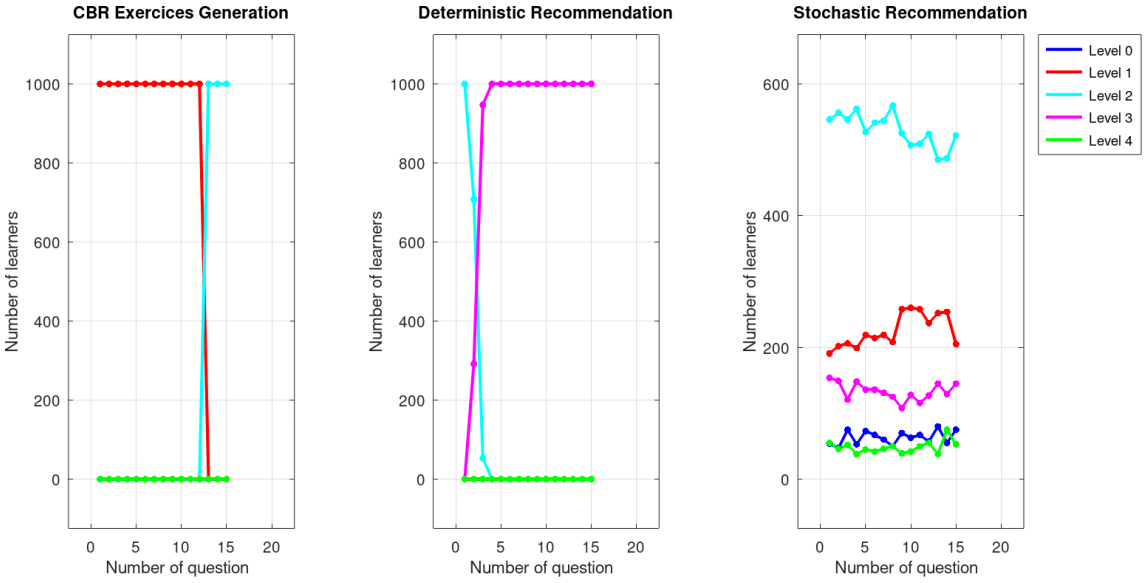

FIGURE 6.4 – Résultats pour le troisième Test

de complexité avec des notes moyennes élevées et si le nombre d'exercices recommandés est trop élevé ou trop faible pour les notes inférieures.

$$rs_c(x) = e^{-\frac{2}{100}(32x_{0,c}^2 - 28x_{0,c} + 10x_{1,c} - 4)^2}; \{x \in \mathbb{R}^2 | 0 <= x <= 1\} \tag{6.12}$$

$$r = \sum_{c=0}^{c_n - 1} rs_c \tag{6.13}$$

Les propriétés de la métrique sont :

— $\{\forall x \in \mathbb{R}^2 | 0 <= x <= 1\}, rs_c(x) > 0$

— $max(rs_c(x)) = 1;\ if\ 16x_{0,c}^2 - 14x_{0,c} + 5x_{1,c} - 2 = 0$

La figure 6.6 montre l'équation globale pour la métrique rs dans le domaine de deux variables $x_{0,c}$ et $x_{1,c}$. La valeur maximale de r dans un niveau de complexité spécifique est de 1, la valeur maximale globale pour les scénarios testés est de 5, un bon système de recommandation doit donc avoir une valeur r élevée.

Les résultats du calcul des métriques pour le système original et les deux modèles dans les trois scénarios définis sont présentés dans le tableau 6.5.

Pour comparer le système original et le modèle de recommandation, est utilisée la métrique de la diversité des propositions, avec la similarité en cosinus (La similarité en cosinus entre un vecteur A et un vecteur B, Équation 6.14) entre toutes les propositions des apprenants. Les résultats de la similarité cosinus moyenne sont présentés dans le tableau 6.6.

$$sc = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}} \tag{6.14}$$

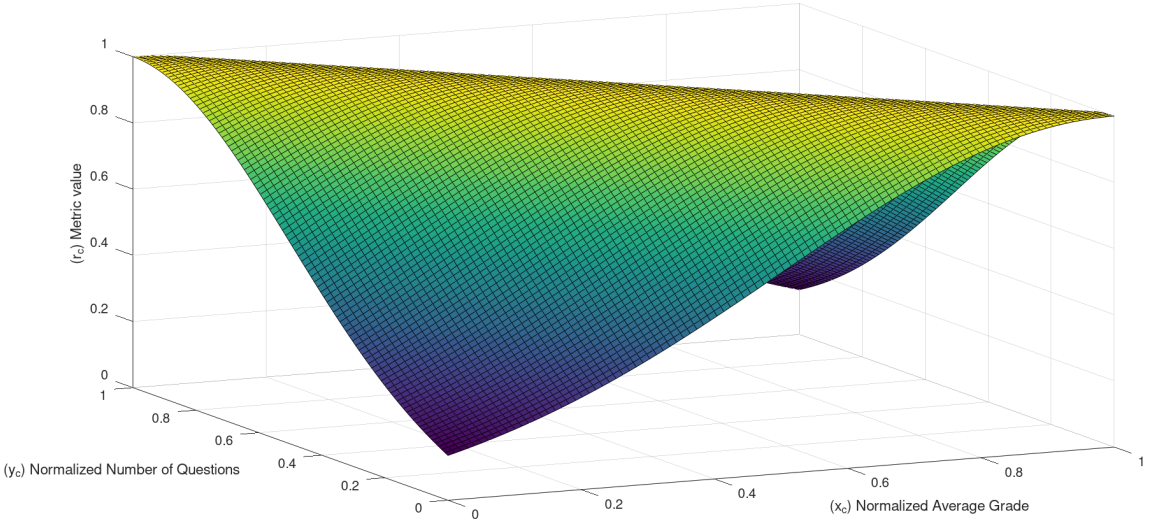

FIGURE 6.5 – Métrique pour le parcours standard

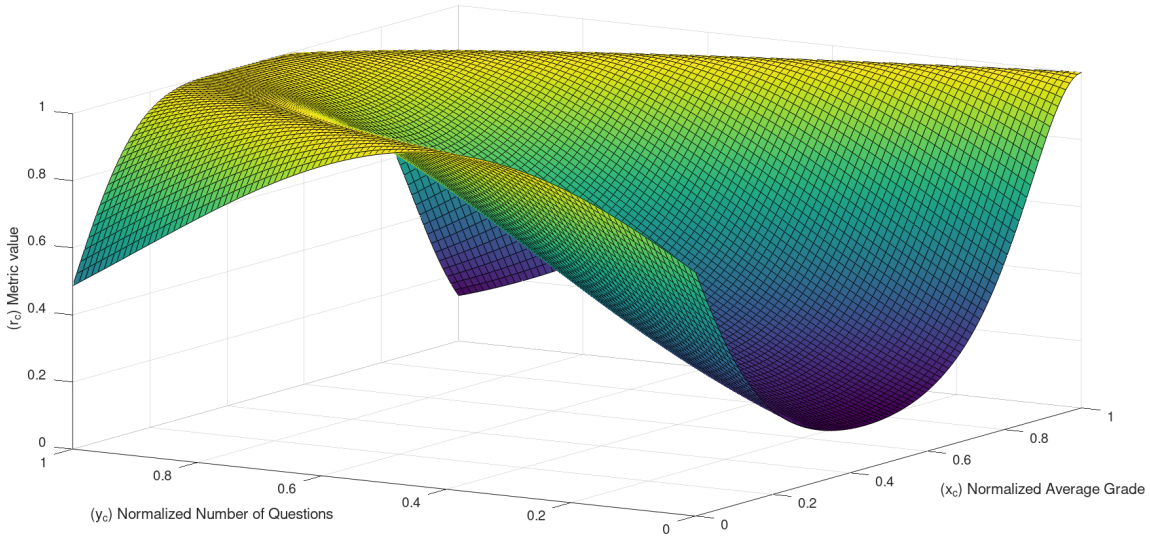

FIGURE 6.6 – Fonction d'évaluation métrique à chaque niveau de complexité (Soft learning)

6.4/ DISCUSSION ET CONCLUSIONS

Avec la génération d'exercices CBR, le système propose les mêmes exercices à tous les apprenants, et l'évolution des niveaux de complexité est très lente, presque un changement toutes les 3 ou 4 sessions, ceci parce que le système ne prend pas en compte les notes obtenues pendant la session. Les systèmes de recommandation sont plus dynamiques et les évolutions sont plus rapides mais en considérant les notes des apprenants, le modèle déterministe suggère des changements de niveaux à un grand nombre d'apprenants soudainement parce qu'ils sont regroupés à l'intérieur d'un intervalle de taux de maîtrise, alors que le modèle stochastique est plus axé sur la personnalisation individuelle et les changements de niveau de complexité sont produits pour un petit nombre d'apprenants. Les deux modèles proposés ont la capacité de détecter les faiblesses des apprenants et d'adapter la session à leurs besoins particuliers.

	c_0	c_1	c_2	c_3	c_4	Total (r)	Total ($\%$)
Test 1							
CBR	**0.9979**	-	-	-	-	0.9979	
DM	0.8994	0.1908	**0.3773**	**0.2990**	-	1.7665	
SM	0.8447	**0.3012**	0.2536	0.2030	**0.1709**	**1.7734**	
Test 2							
CBR	**0.4724**	**0.7125**	-	-	-	1.1849	
DM	-	0.6310	**0.3901**	**0.4253**	-	1.4464	
SM	0.2697	0.7089	0.2634	0.2026	**0.1683**	**1.6129**	
Test3							
CBR	-	**0.9179**	0.2692	-	-	1.1871	
DM	-	-	0.2236	**0.9674**	-	1.191	
SM	0.1873	0.3038	**0.6345**	0.2394	**0.1726**	**1.5376**	

TABLE 6.5 – Résultats de la métrique $rs_c(x)$ (CBR - Système sans modèle de recommandation, DM - Modèle deterministique, SM - Modèle stochastique)

Model	Scenario 1	Scenario 2	Scenario 3
CBR	1	1	1
DM	0.9540	0.9887	0.9989
SM	**0.8124**	**0.8856**	**0.9244**

TABLE 6.6 – Moyenne de la diversité des propositions pour tous les apprenants. Une valeur plus faible représente une plus grande diversité. (CBR - Système sans modèle de recommandation, DM - Modèle deterministique, SM - Modèle stochastique)

La base de données générée a permis de simuler diverses situations avec les notes de 1000 apprenants, permettant ainsi d'évaluer le comportement des systèmes de recommandation avec différentes configurations.

Les résultats numériques utilisant la métrique définie montrent que les distributions des questions dans une session par les deux versions du modèle de recommandation sont différentes mais avec une tendance générale similaire pour tous les apprenants. Le modèle proposé tente de répartir les questions dans tous les niveaux de complexité définis. Globalement, avec la métrique définie, le modèle stochastique a obtenu un meilleur score. Par rapport au système original, le modèle de recommandation (versions déterministe et stochastique) obtient une augmentation globale de l'adaptabilité comprise entre 15% et 68% pour tous les niveaux de complexité.

Selon la métrique de la similarité cosinus, le modèle de recommandation proposé augmente la diversité des propositions par rapport au système original dans les trois scénarios évalués, ce qui indique qu'en plus d'atteindre l'adaptabilité, des propositions personnalisées sont générées tout en maintenant l'objectif de faire progresser les apprenants entre les niveaux de complexité. La diversité des propositions est une caractéristique essentielle du modèle de recommandation dans ses deux versions.

Les modules de recommandation sont une pièce essentielle pour les systèmes ITS car ils aident à guider le processus d'apprentissage individuel, permettent d'identifier les faiblesses et de réorienter le processus complet afin d'améliorer les connaissances et les compétences. Les versions du modèle proposé peuvent détecter en temps réel les faiblesses de l'apprenant et tentent de réorienter la session vers le meilleur niveau de com-

plexité possible afin d'aider l'apprenant à acquérir et à maîtriser les connaissances avant de passer aux niveaux de complexité supérieurs, car généralement les connaissances des niveaux de complexité inférieurs sont nécessaires pour compléter les niveaux supérieurs. Même si l'ensemble de données généré est une simulation des temps de réponse et des notes des apprenants, les tests qui l'utilisent permettent de voir la flexibilité et la robustesse du modèle de recommandation proposé, car les données relatives aux apprenants présentent une grande diversité et obligent le système à s'adapter à différents types de configurations. Par conséquent, il est possible de conclure que le modèle de recommandation proposé a la capacité de fonctionner dans différentes situations et dans chaque cas de proposer des chemins alternatifs pour améliorer le processus d'apprentissage global, même si l'objectif d'apprentissage est différent pour chaque apprenant, comme le démontrent les résultats obtenus dans l'évaluation des deux métriques proposées. Le modèle proposé permet également la diversité et la personnalisation du système, puisque selon les résultats de la comparaison avec la similarité cosinus entre toutes les recommandations générées pour chaque apprenant, il y a une augmentation par rapport au système original.

7

RAISONNEMENT À PARTIR DE CAS (RÀPC) POUR RÉGRESSION

7.1/ INTRODUCTION

Ce chapitre présente un modèle basé sur le raisonnement à partir de cas et les méthodes d'ensemble avec un double empilement itératif en deux étapes pour trouver des solutions approximatives à des problèmes de régression unidimensionnels et multidimensionnels. Une partie du contenu est extrait et traduit de l'article Soto *et al.* [**?**].

Cette approche ne nécessite pas d'entrainement, et peut fonctionner avec des données dynamiques au moment de l'exécution. Les solutions sont générées à l'aide d'algorithmes stochastiques afin de permettre l'exploration de l'espace des solutions, l'évaluation est effectuée en transformant le problème de régression en un problème d'optimisation avec une fonction objectif associée. L'algorithme a été testé en comparaison avec neuf algorithmes de régression classiques sur dix bases de données de régression différentes extraites du site de l'UCI, les résultats montrent que l'algorithme proposé génère dans la plupart des cas des solutions assez proches des solutions réelles, selon le RMSE l'algorithme se trouve globalement parmi les six meilleurs algorithmes et avec MAE au troisième meilleur algorithmes des dix évalués, suggérant que les résultats sont raisonnablement bons.

Les méthodes d'ensemble utilisent plusieurs modèles multiples exécutés indépendamment et leurs résultats sont combinés pour obtenir une prédiction globale finale. L'idée principale est d'améliorer les résultats et la capacité de généralisation des modèles individuels. Certaines méthodes d'ensemble utilisent différents modèles avec différents ensembles de données, d'autres utilisent les mêmes modèles avec des paramètres différents, la combinaison des résultats des multiples modèles peut utiliser différentes stratégies comme des règles simples ou des approches plus complexes [**?**]. Les méthodes d'ensemble sont utiles dans les problèmes de classification et de régression.

Les méthodes d'apprentissage automatique appliquées à la régression permettent de prédire des valeurs pour différents types de problèmes en construisant, évaluant et formant des modèles linéaires et non linéaires complexes, mais s'il est possible d'améliorer la précision en les intégrant, les stratégies d'intégration les plus courantes utilisées pour l'apprentissage d'ensemble sont les suivantes : Stacking, Boosting et Bagging [**?**]. Le Stacking est un type de méta-modèle d'apprentissage profond d'ensemble, dont l'objectif est d'utiliser diverses techniques d'apprentissage automatique pour surmonter les limites

des modèles individuels, l'intégration générant un résultat final avec une précision améliorée [**?**]. Dans les méthodes d'empilage, les algorithmes de base sont appelés niveau-0, il s'agit généralement de modèles ou d'algorithmes d'apprentissage automatique hétérogènes qui travaillent tous avec la même base de données. Le méta-algorithme qui unifie les résultats peut être une autre technique d'apprentissage automatique ou un ensemble de règles qui reçoit comme données d'entrée les résultats des algorithmes de niveau-0 est appelé niveau-1 [**?**].

7.2/ MODÈLE PROPOSÉ

L'algorithme proposé ESCBR (Ensemble Stacking Case-Based Reasoning) est basé sur le paradigme générique CBR avec plusieurs algorithmes de recherche de voisins et de génération de solutions qui ont été intégrés selon une variation du modèle de stacking en deux étapes itératives, cette intégration donne à l'algorithme la capacité de s'adapter à différents types de problèmes, d'éviter les biais et le surentraînement. Les résultats de l'exécution des niveaux de stacking stockent dans la mémoire des conteneurs CBR des informations qui aident à l'apprentissage de l'algorithme au fil des itérations, en plus de faciliter la génération de solutions à divers problèmes sur différentes bases de données sans avoir besoin d'une phase d'entraînement. La conception itérative en deux cycles améliore la capacité de la CBR à travailler et à s'adapter à des problèmes dynamiques en cours d'exécution, comme le montre la figure 8.1.

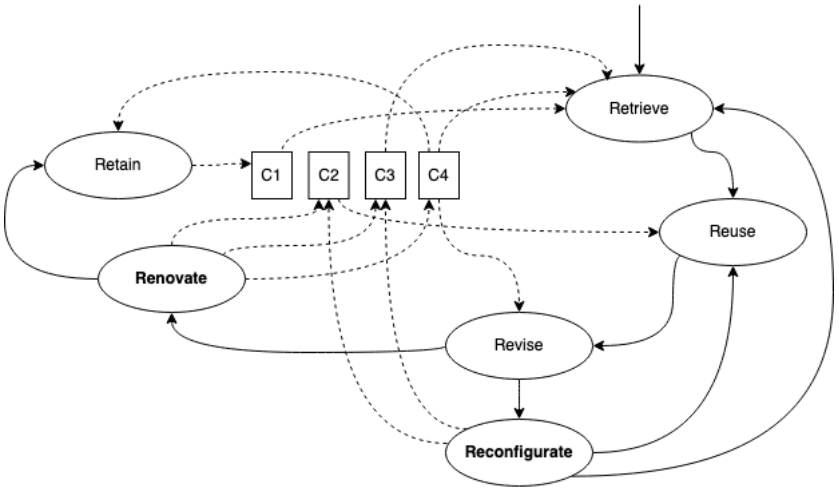

FIGURE 7.1 – Les deux cycles proposés pour le RàPC

L'étape de récupération utilise les algorithmes de recherche et la base de données de cas (conteneurs C1 et C3) pour trouver les voisins les plus proches d'un nouveau problème donné, l'étape de réutilisation utilise les algorithmes de génération de solutions (conteneur C2), l'étape de révision évalue les solutions générées et permet de générer de nouvelles solutions itérativement en fonction des paramètres stockés dans le conteneur C4 et en invoquant l'étape de reconfiguration, avec une solution sélectionnée, l'étape

de renouvellement met à jour les paramètres et les données du conteneur, enfin, dans l'étape de conservation, la base de cas est mise à jour avec le nouveau cas et la solution générée. Le flux complet de l'algorithme proposé est présenté dans la figure 8.2. Les variables et paramètres complets de l'algorithme proposé sont indiqués dans le tableau 8.1.

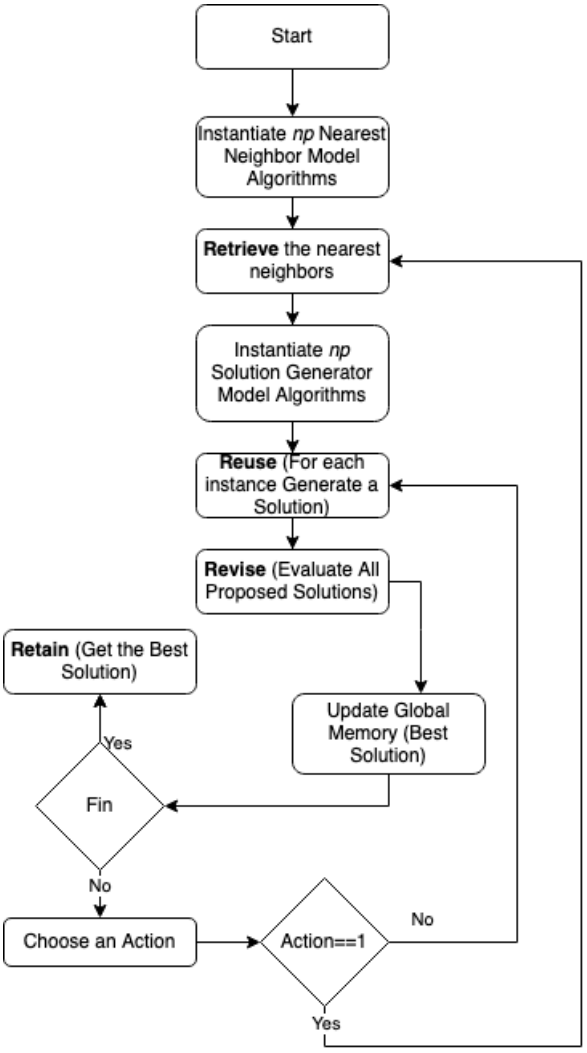

FIGURE 7.2 – Flux du *Stacking* RàPC

7.2.1/ RECHERCHER

La première étape de l'algorithme consiste à trouver les cas les plus similaires à un nouveau cas éventuel, pour cela un premier modèle de stacking avec différents processus est utilisé comme le montre la figure 7.3, au niveau-0 chaque processus sélectionne et exécute un algorithme de recherche de voisins différent choisit parmi r_a modèles dans le conteneur C3, avec un nombre de voisins nl_i choisi aléatoirement dans l'intervalle $[0, nl]$, puis au niveau-1 les résultats sont unifiés en construisant un ensemble global de cas similaires. Cinq algorithmes pour le niveau 0 ont été mis en œuvre pour l'étape de récupération : KNN (K Nearest Neighbors), KMeans, GMM (Gaussian Mixture Model), FuzzyC

ID	Type	Description	Domain
it	p	Nombre d'iterations	$\mathbb{N}, it > 0$
np	p	Nombre de processus	$\mathbb{N}, np > 2$
nl	p	Nombre maximal de voisins locaux	$\mathbb{N}, nl > 0$
ng	p	Nombre de voisins globaux	$\mathbb{N}, ng > 2$
n	v	Dimension de l'espace du problème	$\mathbb{N}, n > 0$
m	v	Dimension de l'espace de solution	$\mathbb{N}, m > 0$
z	v	Taille de la base de données	$\mathbb{N}, z > 0$
p	v	Description du problème	$\mathbb{R}^n$
s	v	Description de la solution	$\mathbb{R}^m$
r_a	v	Nombre de modèles pour l'étape rétrouver	$\mathbb{N}, r_a > 2$
r_b	v	Nombre de modèles pour l'étape de réutilisation	$\mathbb{N}, r_b > 2$
at	v	Identificateur des actions	$[0, 2] \in \mathbb{N}$
nl_i	v	Nombre de voisins locaux pour le modèle i	$\mathbb{N}, nl_i \leq nl$
g	v	Description de la meilleure solution globale	$\mathbb{R}^m$
v	v	Évaluation de la meilleure solution globale	$\mathbb{R}$
$d(x_1, x_2)$	f	Fonction de distance entre x_1 et x_2	$\mathbb{R}$
$MP(x_1^z, x_2, a)$	f	Fonction du modèle pour rétrouver entre x_1 et x_2	$\mathbb{R}^{a \times z}$
$MS(x_1^m)$	f	Fonction du modèle pour réutiliser avec x_1	$\mathbb{R}^m$
$f_s(p^n, s^m)$	f	Évaluation des solutions	$\mathbb{R}$

TABLE 7.1 – Variables et paramètres du modèle proposé (Type : p - paramètre, v - variable, f - fonction)

et KNN Ponderation.

Formellement, le premier modèle de stacking proposé fonctionne avec deux paramètres : une base de données de z cas, un cas est composé de la description du problème et de la description de la solution $(p^n, s^m)^z$ et un nouveau cas sans solution p_w^n, le but de tous les algorithmes de niveau 0 est de générer une liste locale de cas similaires au nouveau cas en utilisant les informations de la description du problème, c'est-à-dire que pour chaque j modèle effectué, est généré l'ensemble $X_j = \{x_1, x_2, ..., x_z \mid x_i = MP_i((p^n)^z, p_w^n, nl_i)\}$. Au niveau 1, un ensemble global est créé à l'aide de tous les ensembles locaux j $X_g = \cup_{n=1}^{ng} min\,;((\cup_{j=1}^{np} X_j) - X_g)$, puis le résultat du premier modèle d'empilement est l'ensemble X_g avec les ng voisins les plus proches.

7.2.2/ RÉUTILISER

Une fois la liste globale des cas similaires établie, les informations correspondant aux solutions de chacun de ces cas sont extraites et utilisées pour générer une nouvelle solution qui s'adapte au nouveau cas en utilisant des cas similaires et des solutions similaires, comme le montre la figure 7.4. La génération est effectuée avec un deuxième modèle de stacking avec différents processus comme le montre la figure 7.5, au niveau 0 chaque processus sélectionne et exécute un algorithme de génération différent à partir des modèles r_b dans le conteneur C2 et au niveau 1 toutes les différentes solutions générées sont stockées dans une mémoire globale. Étant donné que la représentation des solutions est comme un tableau unidimensionnel de multiples cases, alors neuf algorithmes ont été mis en œuvre pour l'étape de réutilisation au niveau 0 : moyenne avec probabilité, moyenne sans probabilité, valeurs médianes, sélection aléatoire avec probabilité, copie

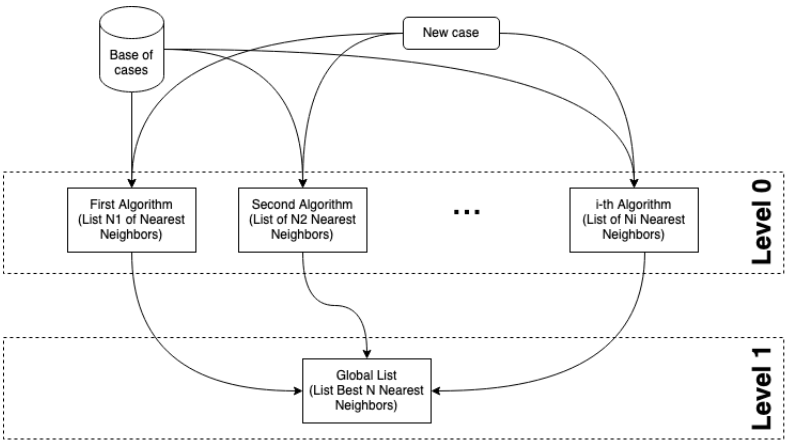

FIGURE 7.3 – *Stacking* pour chercher les plus proches voisins

et changement, vote, interpolation, ACP (analyse en composantes principales) et marche aléatoire. Tous les algorithmes proposées pour générer une nouvelle solution combinent et modifient l'ensemble de solutions construit dans l'étape de récupération.

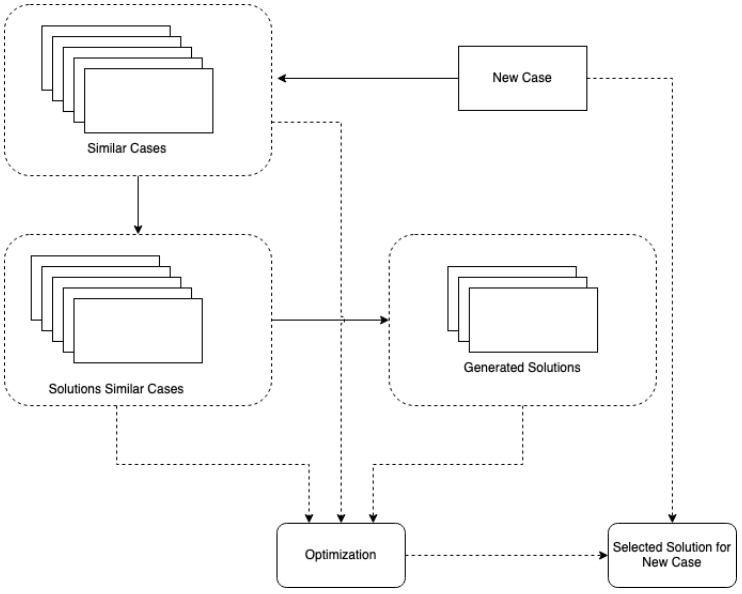

FIGURE 7.4 – Génération et vérification automatique des solutions

La moyenne pondérée avec probabilité permet de construire une solution selon le calcul de la moyenne avec le poids de chaque solution en fonction de sa proximité au nouveau problème. L'équation 7.1 define le calcul de la probabilité en fonction de la distance entre les problèmes et l'équation 7.2 construit la solution en calculant chaque case j.

$$\alpha_j = \frac{d(p_j^n, p_w^n)}{\sum_{i=0}^{nl} d(p_i^n, p_w^n)} \tag{7.1}$$

$$s_{j,w}^m = \frac{1}{nl} \sum_{i=0}^{nl} \alpha_j s_{i,j} \tag{7.2}$$

La moyenne sans probabilité copie aléatoirement les informations des solutions selon la distribution uniforme en donnant la même probabilité à toutes les éléments de toutes les solutions proches. Formellement la génération de la solution est écrite comme l'équation 7.3, où chaque position j est calculée de la même façon.

$$s^m_{w,j} = \frac{1}{nl}\sum_{i=0}^{nl} s_{i,j} \tag{7.3}$$

La génération de la solution par valeurs médianes construit la solution en utilisant la valeur médiane de toutes les solutions pour chaque dimension.

$$s^m_{w,j} = \begin{cases} x_{j,\frac{n+1}{2}}, \ si\ n \in \{2k+1 : k \in \mathbb{Z}\} \\ \frac{x_{j,\frac{n}{2}} + x_{j,\frac{n}{2}+1}}{2}, \ autrement \end{cases} \tag{7.4}$$

La sélection aléatoire avec probabilité génère une solution en copiant aléatoirement l'information des solutions dans chaque position ayant une plus grande probabilité vers le cas de problème associé le plus proche.

$$s^m_{w,k} = max(P(S_j))_k \ with\ probability\ \alpha_j\ \forall k = \{1, ..., m\} \tag{7.5}$$

complete all the models formalization...

Copie/changement, copie les informations d'une solution aléatoire et change une partie avec les informations d'une autre solution sélectionnée aléatoirement.

$$s^m_w = max(P(S_j))\ with\ uniform\ probability \tag{7.6}$$

Le vote permet de copier l'information qui se présente avec plus de fréquence entre toutes les solutions.

$$s^m_{w,j} = max(\#(s_j)) \tag{7.7}$$

L'interpolation utilise des informations aléatoires à partir d'une fonction d'interpolation lineaire 1D calculée.

$$s^m_{w,j} = rand\left(\left(\frac{y_i - y_{i+1}}{x_i - x_{i+1}}\right)(x - x_i) + y_i\right), \forall i \mid 0 \leq i < nl \tag{7.8}$$

Avec PCA, la description du problème est transformée dans l'espace de description des solutions pour établir une relation entre le problème et sa solution, avec le problème et la solution dans le même espace est calcule la moyenne de la distance entre tous les paires problème-solution et en utilisant la description du nouveau problème et la valeur moyenne de distance une solution est générée.

$$s^m_{w,j} = \tag{7.9}$$

Marche aléatoire choisit une solution et change les valeurs dans une dimension aléatoirement avec un petit pas. Le petit pas est une valeur aléatoire générée avec une distribution de probabilité normal.

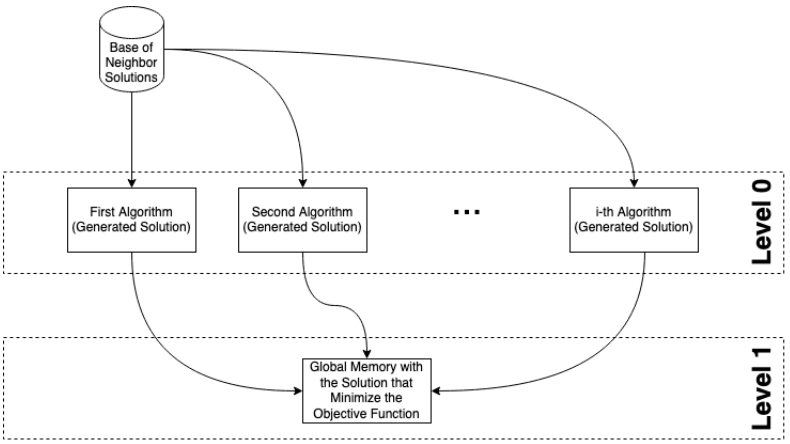

FIGURE 7.5 – *Stacking* pour la génération de solutions

$$s^m_{w,j} = \begin{cases} s^m_j + randNormal(0,1) & if\ randIntU(0,10)\%2 = 0 \\ s^m_j - randNormal(0,1) & otherwise \end{cases} \tag{7.10}$$

****

Expliciter chacune des méthodes de génération des solutions.

****

Le deuxième modèle de stacking fonctionne avec la description des solutions s comme paramètre, avec l'ensemble $(s^m)^{ng}$, chaque modèle de réutilisation exécuté peut générer une solution candidate $s_{i,c} = MS_i((s^m)^{ng})$. Le niveau 1 est la construction de l'ensemble d'unification de toutes les solutions candidates $Y_g = \cup_{i=1}^{np} s_{i,c}$. Cet ensemble est évalué à l'aide d'une fonction permettant de déterminer la qualité de la solution.

7.2.3/ RÉVISION

Dans cette phase, le problème de l'évaluation automatique d'une solution candidate est transformé en un problème d'optimisation, où la fonction objective est 7.12. Ce problème est similaire au probleme de trouver la moyenne geometrique ou problème de "Fermat-Weber", dans le cas de un espace multidimensionnel, le problème est connu comme moyenne spatiale [**?**]. La figure 7.6 montre un exemple de la formulation du problème en deux dimensions, où le point rouge représente une posible solution qui minimise la somme des distances avec tous les points définis dans l'espace.

La fonction objectif 7.12 établi un rapport entre la distance de la solution générée s^m_w et les x solutions connues s^m_x avec un facteur aléatoire de *drift* ainsi que la distance du problème nouveau p^n_w et les x problèmes connus p^n_w. Ici, la complexité de trouver le point optimal se trouve dans le fait que tous les points dans l'espace ne sont pas valides comme solution du nouveau problème, c'est la raison pour laquelle le point doit être généré en utilisant l'information des solutions connues. L'objectif de faire ça est de utiliser l'information disponible qui décrit un problème et sa solution pour valider et générer l'ensemble de solutions proposées.

$$\lambda_x(p_w, s_w) = \left(\frac{d(s_w^m, (s_x^m + rn(0, d(p_w^n, p_i^n))))}{d(p_w^n, p_x^n)^2} \right) \tag{7.11}$$

$$min \ (f_s(p_w^n, s_w^m)) = min \left(\sum_{i=1}^{ng} \lambda_i(p_w, s_w) \right) \tag{7.12}$$

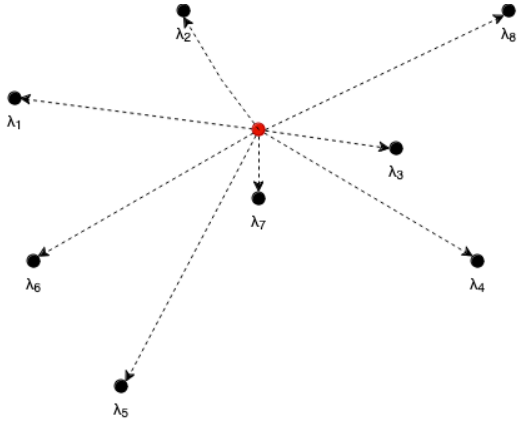

FIGURE 7.6 – Représentation graphique en deux dimensions du problème de moyenne geométrique. (Points associés au problème $(\lambda_1, .., \lambda_7)$ et point rouge solution au problème)

Le cycle d'optimisation peut exécuter les phases de récupération et de réutilisation en fonction de l'action aléatoire sélectionnée parmi $[0, at]$ à chaque itération it, en sauvegardant dans une mémoire globale à chaque itération la solution qui obtient la valeur minimale dans l'évaluation de la fonction objective.

7.2.4/ MÉMORISATION

L'étape de rétention consiste simplement à prendre la meilleure solution proposée et à déterminer s'il s'agit d'une solution nouvelle ou existante et, si elle est nouvelle, à l'enregistrer dans la base de données.

7.3/ RÉSULTATS

Afin de comparer les performances de prédiction et le comportement de l'algorithme proposé, dix bases de données de régression présentant des caractéristiques différentes ont été sélectionnées. Les bases de données et leurs caractéristiques sont indiquées dans le tableau 8.3. Les valeurs des paramètres de l'algorithme sont les suivantes : $it = 100$, $np = 50$, $nl = 10$ et $ng = 10$.

L'algorithme proposé est comparé à neuf algorithmes de régression bien connus et largement utilisés dans divers travaux de recherche et problèmes appliqués. La liste des algorithmes est présentée dans le tableau 9.3. Tous les algorithmes ont été exécutés 100 fois, et les algorithmes qui nécessitent un entraînement et des validations croisées sont exécutés avec $k = 10$.

ID	DataSet	Features	Instances	Output Dimension	Input Domain	Output Domain
DS1	Yatch Hydrodynamics	6	308	1	$\mathbb{R}$	$\mathbb{R}$
DS2	Electrical Grid Stability	12	10000	1	$\mathbb{R}$	$\mathbb{R}$
DS3	Real State Valuation	6	414	1	$\mathbb{R}_+$	$\mathbb{R}_+$
DS4	Wine Quality (Red)	11	1598	1	$\mathbb{R}_+$	$\mathbb{N}$
DS5	Wine Quality (White)	11	4897	1	$\mathbb{R}_+$	$\mathbb{N}$
DS6	Concrete Compressive Strength	8	1030	1	$\mathbb{R}_+$	$\mathbb{R}_+$
DS7	Energy Efficiency	8	768	2	$\mathbb{R}_+$	$\mathbb{R}_+$
DS8	Gas Turbine CO, NOx Emission (2015)	9	7384	2	$\mathbb{R}_+$	$\mathbb{R}_+$
DS9	Student Performace Portuguese	30	649	3	$\mathbb{N}*$	$\mathbb{N}$
DS10	Student Performance Math	30	395	3	$\mathbb{N}*$	$\mathbb{N}$

TABLE 7.2 – Description des bases de données évaluées. (* après codification comme *String*)

ID	Algorithm	ID	Algorithm
A1	Linear Regression	A6	Polinomial Regression
A2	K-Nearest Neighbor	A7	Ridge Regression
A3	Decision Tree	A8	Lasso Regression
A4	Random Forest (Ensemble)	A9	Gradient Boosting (Ensemble)
A5	Multi Layer Perceptron	A10	Proposed Case Based Reasoning

TABLE 7.3 – Liste des algorithmes évalués

Le tableau 8.2 présente le classement moyen de toutes les bases de données en fonction de l'erreur quadratique moyenne (RMSE). Les résultats détaillés des mêmes algorithmes et des mêmes bases de données, mais comparés avec la métrique MAE (Median Absolute Error), sont présentés dans le tableau 8.5.

Dataset	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
DS1	9.010	10.780	1.224	0.982	3.369	9.009	8.985	9.629	**0.668**	5.871
DS2	0.022	0.025	0.020	0.012	0.017	0.022	0.022	0.037	**0.011**	0.015
DS3	8.633	8.033	9.334	**7.203**	8.470	8.705	8.842	9.009	7.324	8.491
DS4	0.651	0.746	0.782	**0.571**	0.694	0.651	0.651	0.792	0.617	0.762
DS5	0.753	0.806	0.820	**0.599**	0.853	0.754	0.757	0.863	0.688	0.748
DS6	10.439	8.871	6.144	**4.738**	6.553	10.423	10.422	10.428	5.053	8.766
DS7	2.948	2.116	0.541	**0.465**	3.726	2.949	2.979	4.094	0.467	1.973
DS8	1.315	1.161	1.513	**1.109**	1.566	1.303	1.308	1.318	1.125	2.157
DS9	**2.304**	2.624	3.217	2.315	2.898	**2.304**	**2.304**	2.551	2.342	2.802
DS10	3.052	3.404	4.158	**3.014**	3.607	3.061	3.061	3.150	3.020	3.874
Avg. Rank	5.7	6.3	7.2	2.1	6.6	5.6	5.5	8.6	1.8	5.6

TABLE 7.4 – Résultats de la métrique RMSE (Root Mean Squared Error) pour les bases de données évaluées avec des algorithmes de régression

Dataset	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
DS1	6.776	2.385	0.231	0.207	3.632	6.778	6.307	5.186	**0.162**	1.193
DS2	0.015	0.017	0.012	0.008	0.012	0.015	0.015	0.030	**0.007**	0.011
DS3	5.092	4.320	4.1	3.632	4.435	5.092	5.20	5.132	**3.504**	3.90
DS4	0.413	0.495	0.18	0.325	0.451	0.413	0.412	0.544	0.387	**0.154**
DS5	0.509	0.548	0.285	0.374	0.550	0.509	0.509	0.633	0.456	**0.113**
DS6	6.989	5.709	3.134	**2.839**	4.306	6.989	6.989	6.986	3.084	5.439
DS7	1.393	1.372	**0.217**	0.218	2.523	1.393	1.529	2.346	0.243	1.008
DS8	0.549	0.297	0.365	**0.289**	0.742	0.549	0.549	0.540	0.309	0.861
DS9	**1.496**	1.788	2.080	1.612	2.005	**1.496**	**1.496**	1.714	1.538	1.721
DS10	2.344	2.534	2.910	2.331	2.543	2.344	2.344	2.481	**2.258**	2.602
Avg. Rank	6.45	6.4	4.35	2.3	7.35	6.55	6.6	7.9	2.4	4.7

TABLE 7.5 – Résultat de la métrique MAE (Median Absolute Error) pour les bases de données évaluées avec des algorithmes de régression

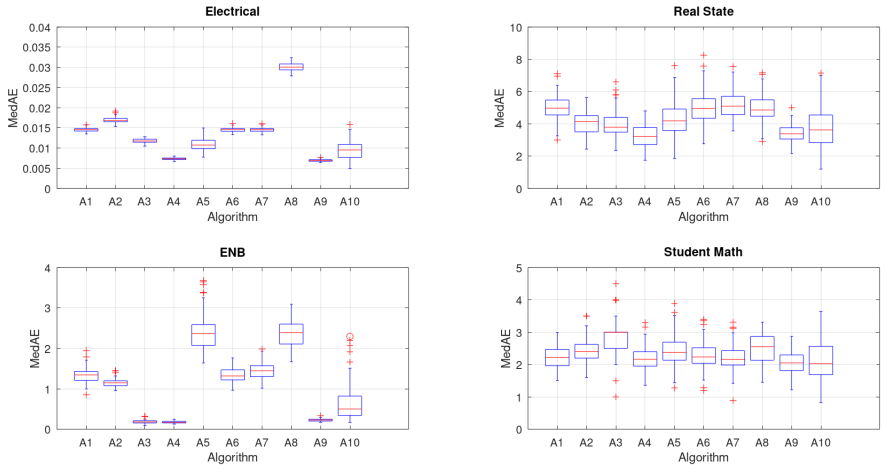

FIGURE 7.7 – Résultats de la métrique MAE (*Median Absolute Error*) pour les dix algorithmes et quatre bases de données représentatives

La dispersion globale, la médiane et les valeurs aberrantes pour quatre bases de données représentatives sont présentées dans la figure 8.5, où l'on peut voir que l'algorithme proposé génère plus de valeurs aberrantes que les autres algorithmes, mais la variance est faible et la convergence est proche de la valeur réelle, meilleure que la plupart des algorithmes comparés.

7.4/ DISCUSSION

L'algorithme proposé révèle une performance compétitive par rapport à certains des algorithmes les plus populaires, les plus utilisés et les plus récents pour la prédiction dans les problèmes de régression. En particulier, dans ce travail, nous avons effectué les tests sur dix bases de données avec différentes caractéristiques, telles que : le nombre d'instances, le nombre de caractéristiques, le domaine des variables d'entrée, les dimensions de la variable de sortie et le domaine d'application. Cela démontre la polyvalence de l'algorithme proposé et son applicabilité à différentes configurations. Étant donné la nature exploratoire et stochastique de l'algorithme proposé, il présente une grande diversité de solutions générant plusieurs valeurs aberrantes, mais malgré cela, dans la plupart des cas, il est possible d'atteindre une solution approximative qui converge vers la solution réelle, c'est la raison pour laquelle, dans certains cas, les valeurs de la moyenne sont élevées, mais celles de la médiane restent faibles.

On constate également que l'intégration des algorithmes de recherche produit de meilleurs résultats que les algorithmes simples, comme dans le cas de l'algorithme proposé par rapport au KNN ou par rapport à la régression linéaire, bien que pour l'algorithme proposé, l'impact du premier et du deuxième empilement sur les résultats finaux obtenus n'ait pas été déterminé avec précision.

Globalement, pour le RMSE, les algorithmes de boosting sont plus performants que les algorithmes classiques, même si les performances sont variables. L'algorithme proposé obtient des valeurs acceptables dans toutes les bases de données. D'après la moyenne

des positions de classement, pour le RMSE, il est placé à la sixième place, pour le MAE, l'algorithme obtient la meilleure valeur dans trois des dix bases de données et il est placé globalement à la troisième place.

Un aspect important de l'algorithme proposé est la fonction objective, qui peut être évaluée et modifiée dynamiquement en fonction des caractéristiques du problème évalué, étant donné que dans la présente étude les tests ont été effectués avec la fonction intuitive qui fournit une plus grande probabilité de sélection et d'évolution à la solution associée aux voisins les plus proches, mais il est possible de compléter l'évaluation avec d'autres termes pertinents et d'améliorer ainsi les résultats.

Outre les résultats, l'algorithme proposé présente plusieurs avantages par rapport aux algorithmes avec lesquels il a été comparé, parmi lesquels : il ne nécessite pas d'entraînement, il peut intégrer des algorithmes et des règles dans chaque pile et, grâce à la conception en deux cycles, il peut travailler avec des problèmes dynamiques en cours d'exécution et la variance est faible dans les résultats produits.

7.5/ CONCLUSION

Ce chapitre propose une technique de régression générique utilisant le raisonnement basé sur les cas et le modèle d'empilement, dont les principales caractéristiques sont qu'elle ne nécessite pas de formation et que, grâce au cycle itératif interne, elle peut s'adapter à des problèmes dynamiques en temps réel. Les résultats numériques obtenus lors des tests effectués montrent le potentiel de l'algorithme avec des données variées et des bases de données de différentes tailles ainsi que la compétitivité avec d'autres algorithmes standard et robustes couramment utilisés dans les problèmes de régression.

8

Intégration du Raisonnement à partir de cas (RàPC) et des Systèmes Multi-Agent (SMA)

8.1/ Introduction

Ce chapitre montre comme a été conçu et implémenté un algorithme pour intégrer l'algorithme de raisonnement à partir de cas explicité dans le chapitre précedent avec les systèmes multi-agents pour améliorer la performance et les résultats pour traiter des problèmes de régression.

Un système multi-agents est un système décentralisé composé de plusieurs entités appelées agents qui ont la capacité d'effectuer des actions spécifiques et de réagir à l'environnement en fonction des informations partielles qu'ils peuvent obtenir, ils peuvent également collaborer les uns avec les autres et coopérer pour atteindre un objectif spécifique. À partir des informations qu'ils perçoivent et échangent, les agents peuvent apprendre de manière autonome et atteindre leurs objectifs de manière efficace [**?**]. Les systèmes multi-agents peuvent être exécutés en parallèle étant donné l'indépendance des agents, ils sont robustes aux problèmes qui présentent une incertitude [**?**].

Le raisonnement Bayésien est une hypothèse sur le fonctionnement du cerveau humain et sa relation avec l'environnement, dont le principe est que les expériences passées permettent de déduire les états futurs, ce qui permet d'agir et de prendre des décisions [**?**], également des informations peuvent être deduites et d'apprendre à partir d'informations incomplètes [**?**].

Nous proposons ici un modèle qui intègre un algorithme de raisonnement à paritr de cas avec des systèmes multi-agents cognitifs qui utilisent le raisonnement Bayésien afin d'améliorer les processus de recherche de voisins et de génération de solutions en utilisant l'échange d'informations inter-agents et l'apprentissage par renforcement, obtenant ainsi des effets émergents dans le comportement global de l'algorithme, ce qui peut conduire à l'amélioration des prédictions de l'algorithme ESCBR [**?**].

8.2/ MODÈLE PROPOSÉ

L'algorithme ESCBR-SMA est basé sur le paradigme RàPC générique et incorpore divers algorithmes de recherche de voisins et de génération de solutions. Ceux-ci sont intégrés à l'aide d'une variante du modèle d'empilement en deux étapes itératives, exécutées par des agents qui exploitent indépendamment les algorithmes définis dans les conteneurs de révision et de réutilisation. Les agents possèdent une mémoire locale qui leur permet d'ajuster et de développer leur recherche de voisins, en générant des solutions à chaque itération. Chaque agent individuel a un comportement correspondant qui permet l'échange d'informations entre les autres agents. Le processus de décision et les étapes des agents au sein du système élargi sont décrits dans la figure 8.1.

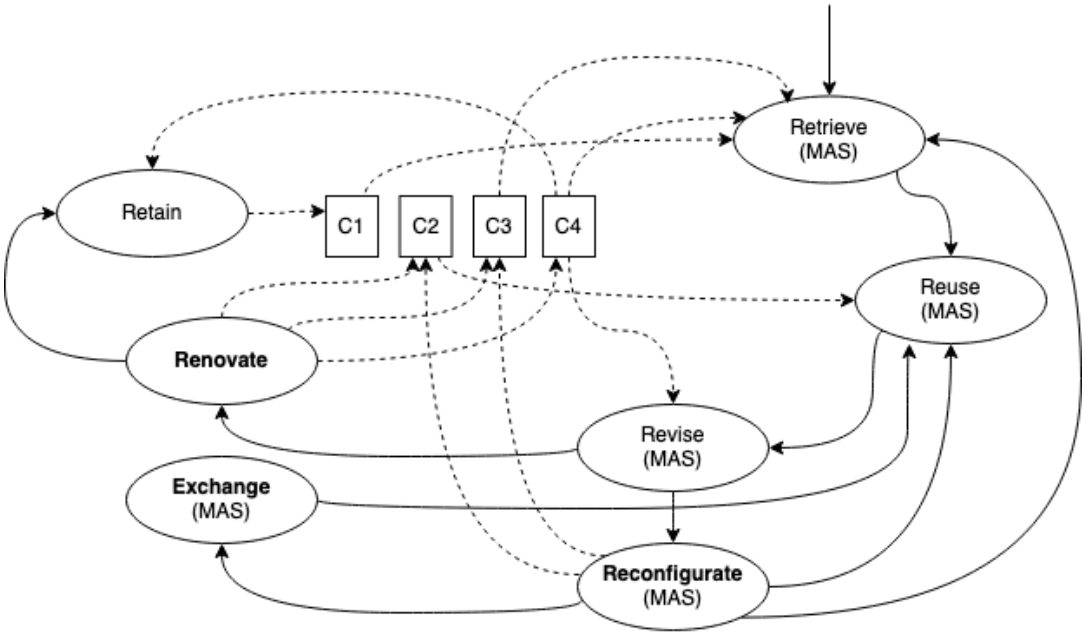

FIGURE 8.1 – Deux cycles du système ESCBR-SMA proposé

ESCBR-SMA correspond aux deux étapes de l'ESCBR (Ensemble Stacking Case-Based Reasoning) intégrées à un système multi-agents. Chaque agent effectue un algorithme de recherche des voisins du problème au cours de la première étape et, au cours de la seconde, génère une solution en se référant à la liste des solutions obtenues au cours de la première étape. À chaque itération, les agents peuvent choisir parmi trois comportements programmés : la recherche de problèmes voisins, la génération de solutions ou l'échange d'informations avec un autre agent. L'exécution asynchrone est employée pour ces comportements, tandis que la création et la sélection de la liste des voisins se font de manière synchrone pour unifier l'algorithme global. La création et la sélection des voisins se font de manière synchrone pour unifier l'algorithme global.

Les étapes d'extraction, de réutilisation, de révision et de conservation restent conformes au modèle RàPC conventionnel. L'algorithme proposé comprend désormais trois nouvelles étapes. Dans la phase de reconfiguration, les agents mettent à jour les valeurs de

leurs paramètres locaux afin d'améliorer la qualité de la solution proposée lors de l'itération suivante. Dans la phase d'échange, les agents échangent des informations pour améliorer leurs processus internes de recherche et de génération. L'étape de rénovation met à jour les valeurs des paramètres globaux de l'algorithme. Le flux complet de l'algorithme proposé est présenté dans la figure 8.2, où l'on peut voir que l'algorithme commence par la création de n agents. Lors de l'initialisation, les agents sélectionnent au hasard un algorithme de récupération et un algorithme de réutilisation, et initialisent également les vecteurs bayésiens correspondants. Tous les agents travaillent avec le même ensemble de données. En parallèle, les agents exécutent l'algorithme de récupération sélectionné, à partir des problèmes similaires trouvés, chaque agent extrait les solutions et exécute l'algorithme spécifique de génération d'une nouvelle solution. Toutes les solutions proposées par les agents sont évaluées à l'aide d'une fonction objective de minimisation, c'est-à-dire que la solution qui minimise la fonction objective est la meilleure solution possible. Au cours de l'étape suivante, les agents sélectionnent au hasard un algorithme de récupération et un algorithme de réutilisation, puis mettent à jour les vecteurs bayésiens en fonction des résultats obtenus avec la fonction objective. Lors de l'itération suivante, chaque agent peut choisir l'une des trois actions possibles : changer les algorithmes de récupération et de réutilisation, ne changer que l'algorithme de réutilisation ou échanger des informations avec un autre agent choisi au hasard. Le choix aléatoire d'une action différente permet à l'agent de construire et d'améliorer une solution candidate.

Les variables et paramètres complets de l'algorithme proposé sont présentés dans le tableau 8.1. Le système multi-agents est composé d'un nombre variable d'agents, tous homogènes mais dotés de processus cognitifs internes indépendants et stochastiques.

ID	Type	Description	Domain
it	p	Nombre d'itérations	$\mathbb{N}, it > 0$
np	p	Nombre d'agents	$\mathbb{N}, np > 2$
nl	p	Nombre maximal de voisins locaux	$\mathbb{N}, nl > 0$
ng	p	Nombre de voisins globaux	$\mathbb{N}, ng > 2$
n	v	Dimension de l'espace du problème	$\mathbb{N}, n > 0$
m	v	Dimension de l'espace de solution	$\mathbb{N}, m > 0$
p	v	Description du problème	$\mathbb{R}^n$
s	v	Description de la solution	$\mathbb{R}^m$
p_w	v	Description du nouveau problème	$\mathbb{R}^n$
s_w	v	Description de la nouvelle solution	$\mathbb{R}^m$
n_{rt}	v	Nombre d'algorithmes por l'étape de rétrouver	$\mathbb{Z}$
n_{rs}	v	Nombre d'algorithmes de réutilisation	$\mathbb{Z}$
$d(x_1, x_2)$	f	Fonction de distance entre x_1 et x_2	$\mathbb{R}$
$rn(x, y)$	f	Valeur aléatoire avec distribution normale x moyenne, y écart-type	$\mathbb{R}_+$
$rnp(x, y)$	f	Valeur aléatoire discrète, x nombre d'options y vecteur discret de probabilités	$\mathbb{Z}$
$f_s(p^n, s^m)$	f	Évaluation des solutions	$\mathbb{R}$

TABLE 8.1 – Variables et paramètres du modèle proposé (Type : p - paramètre, v - variable, f - fonction)

8.2.1/ ALGORITHMES

Lors de la première étape de l'empilage, chaque agent peut sélectionner l'un des algorithmes mis en œuvre pour rechercher des problèmes similaires au nouveau problème

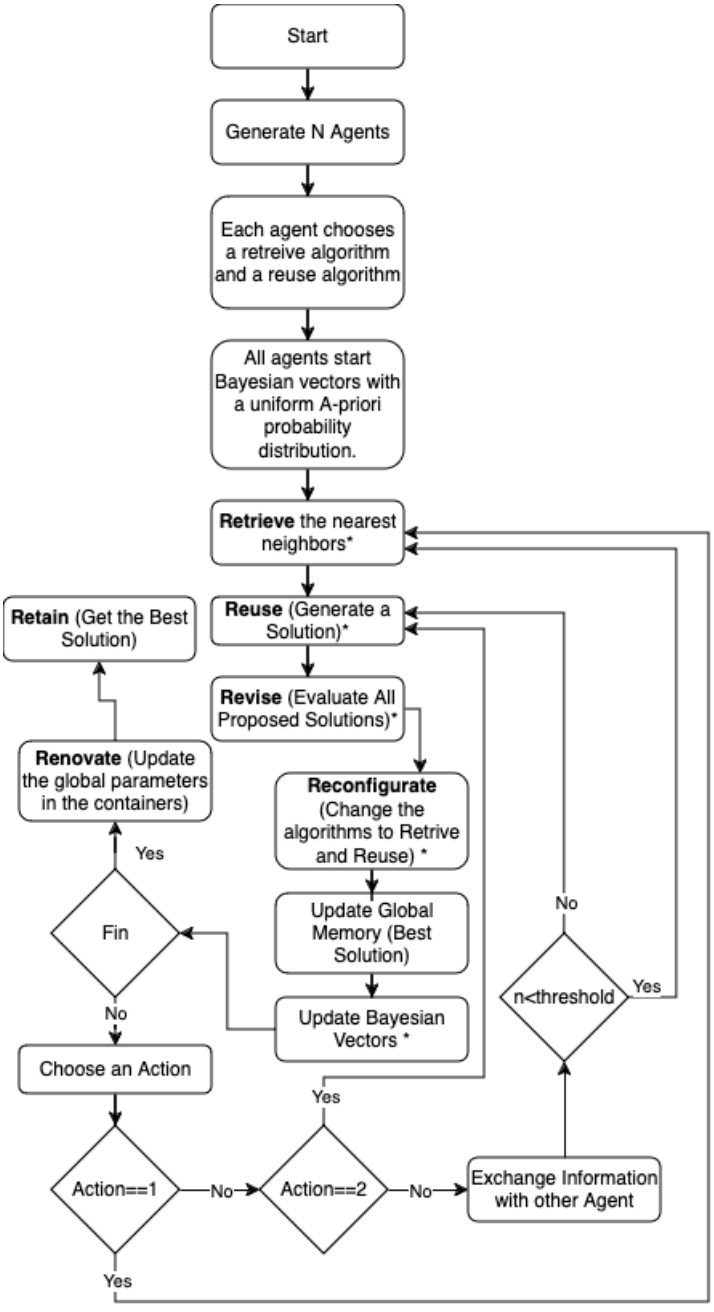

FIGURE 8.2 – Flux du *Stacking* du RàPC (* est une tâche effectuée par chaque agent)

posé. Les algorithmes sont les suivants : KNN (K-Nearest Neighbor), KMeans, GMM (Gaussian Mixture Model), FuzzyC et KNN pondéré.

Lors de la deuxième étape de l'empilage, les agents peuvent choisir un algorithme parmi les suivants : pondération avec probabilité, pondération sans probabilité, valeurs médianes, Copier/Changer, vote, interpolation, PCA (analyse en composantes principales) et pas aléatoire.

La sélection aléatoire pondérée avec l'algorithme de probabilité construit une solution en copiant aléatoirement les informations des solutions ayant une probabilité plus élevée vers le cas de problème associé le plus proche. En revanche, la sélection aléatoire pon-

dérée sans algorithme de probabilité copie aléatoirement les informations des solutions selon la distribution uniforme. Les valeurs médianes utilisent la valeur médiane de toutes les solutions pour chaque dimension. L'algorithme basé sur la copie/le changement copie les informations d'une solution aléatoire et en modifie une partie avec les informations d'une autre solution sélectionnée de manière aléatoire. L'algorithme basé sur le vote copie l'information la plus fréquente entre les solutions. L'algorithme basé sur l'interpolation utilise des informations aléatoires à partir d'une fonction d'interpolation calculée. Le générateur avec PCA transforme la description du problème en un espace de descriptions de solutions afin d'établir une relation entre le problème et sa solution. Enfin, l'algorithme de pas aléatoire sélectionne une solution et modifie les valeurs dans une seule dimension de manière aléatoire avec un petit pas dans l'espace des solutions.

8.2.2/ STRUCTURE DES AGENTS

La structure des agents est similaire pour tous, mais chaque agent effectue un processus cognitif individuel qui lui permet d'obtenir un comportement indépendant et différent de tous les autres. La figure 8.3 montre les actions et les variables nécessaires à l'exécution de l'ensemble du processus.

Il y a trois actions possibles à exécuter (« Récupérer et réutiliser », « Réutiliser » et « Échanger ») et huit variables : le nombre de voisins qui définit le nombre de problèmes similaires au nouveau problème posé que l'agent doit rechercher dans la base de données, la liste des voisins les plus proches établit la liste des agents voisins avec lesquels des informations peuvent être échangées, l'algorithme de récupération est l'identifiant de l'algorithme que l'agent va exécuter pour trouver les problèmes similaires au nouveau problème, l'algorithme de réutilisation est l'identifiant de l'algorithme que l'agent va exécuter pour générer une solution candidate au nouveau problème, solution est la description de la solution générée, évaluation de la solution est la valeur numérique décrivant la qualité de la solution générée selon une fonction d'optimisation (eq 8.1 et eq 8.2), vecteur bayésien pour les algorithmes de récupération contenant les valeurs de probabilité pour chacun des algorithmes de récupération et vecteur bayésien pour les algorithmes de réutilisation contenant les valeurs de probabilité pour chacun des algorithmes de réutilisation.

$$min\ \ (f_s(p_w^n, s_w^m)) = min\left(\sum_{i=1}^{ng} \frac{d(s_w^m, s_i^t)}{d(p_w^n, p_i^n)^2}\right) \tag{8.1}$$

$$s_i^t = s_i^m + rn(0, d(p_w^n, p_i^n)) \tag{8.2}$$

8.2.3/ APPRENTISSAGE DES AGENTS

Tous les agents contiennent un vecteur bayésien de probabilité pour la phase de récupération (conteneur C3) et un vecteur bayésien de probabilité pour la phase de réutilisation (conteneur C2). Initialement, ces vecteurs sont configurés avec une probabilité uniforme pour tous les algorithmes de récupération et de réutilisation, il s'agit de la distribution de probabilité *a priori*, mais pour chaque itération, les vecteurs sont mis à jour avec l'équation bayésienne en utilisant les meilleurs résultats du même agent comme paramètre

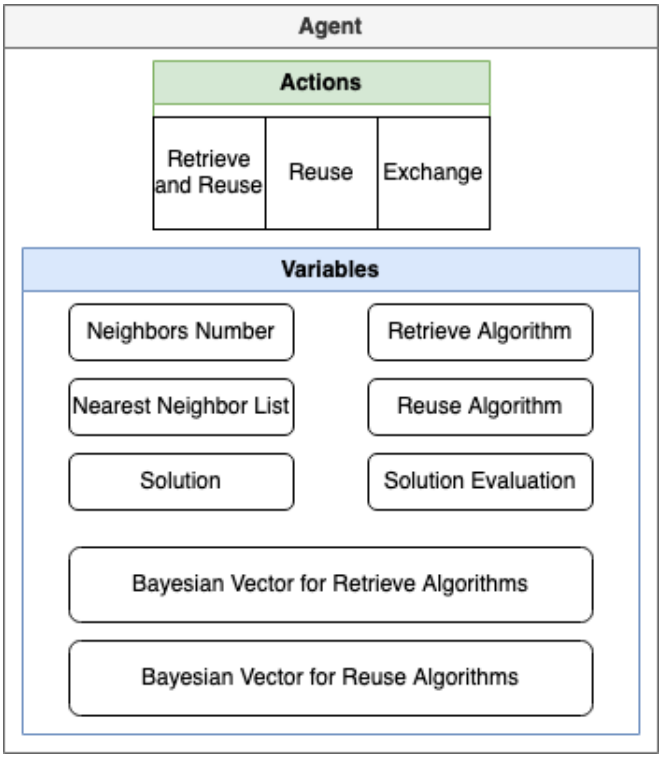

FIGURE 8.3 – Structure interne des agents

de vraisemblance. Le choix suivant d'algorithmes pour la récupération et la réutilisation calcule la probabilité avec les vecteurs. L'agent apprend ainsi de son expérience et sélectionne les algorithmes les plus susceptibles de fournir les meilleures réponses au regard de l'objectif défini.

L'apprentissage est basé sur un raisonnement bayésien où les vecteurs de récupération et de réutilisation sont initialisés avec une probabilité uniforme pour tous les algorithmes, comme le montre la figure 8.4. Au cours d'une itération, si un algorithme a contribué à la construction de la meilleure solution, il reçoit un point pour chaque agent qui l'a utilisé. Ces informations sont stockées dans un vecteur normalisé qui est ensuite utilisé comme vecteur de vraisemblance $P(A)$ pour calculer les nouvelles probabilités dans l'équation bayésienne 8.3, $P(B)$ est le terme de normalisation global.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{8.3}$$

La sélection d'un algorithme de recherche a_{rt} se fait au moyen d'une valeur aléatoire tirée du vecteur de distribution de recherche discrète obtenu par raisonnement bayésien, comme le montre l'équation 8.4.

$$a_{rt} = rnp(n_{rt}, P(A|B)_{rt}) \tag{8.4}$$

La sélection d'un algorithme de réutilisation a_{rs} se fait au moyen d'une valeur aléatoire tirée du vecteur de distribution de réutilisation discrète obtenu par raisonnement bayésien, comme le montre l'équation 8.5.

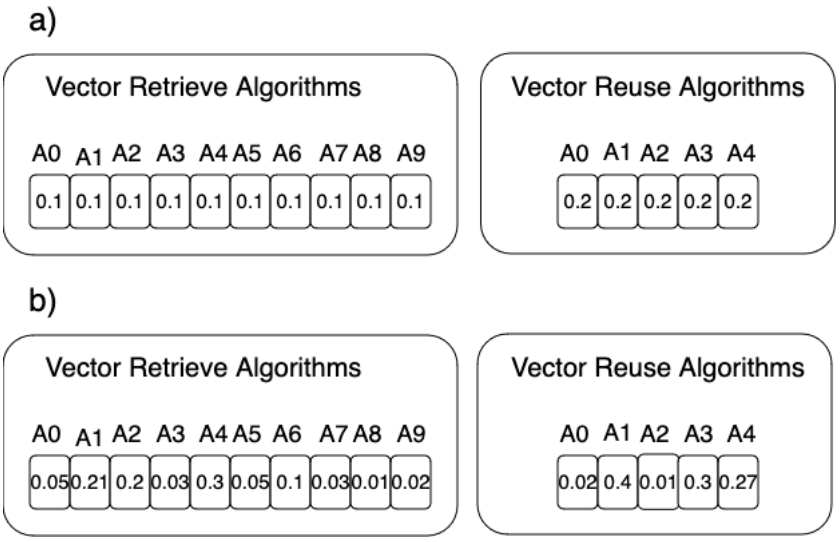

FIGURE 8.4 – Exemple d'évolution Bayésienne des vecteurs pour un agent. a) Initialisation des probabilités $P(B|A)$ vecteurs pour Retrieve et Reuse, b) Probabilités après quelques itérations $P(A|B)$ vecteurs pour Retrieve et Reuse

$$a_{rs} = rnp(n_{rs}, P(A|B)_{rs}) \tag{8.5}$$

Le processus d'apprentissage est réalisé individuellement par chaque agent, mais comporte un aspect collaboratif puisque les agents communiquent les informations qu'ils ont trouvées, les optimisations locales qu'ils ont réalisées, ainsi que les algorithmes et les paramètres.

8.2.4/ ÉCHANGES ENTRE LES AGENTS

Les agents peuvent modifier leurs informations au cours d'une itération en choisissant au hasard un voisin dans leur liste de voisins les plus proches. Les changements permettent d'obtenir une grande probabilité de propager les paramètres des agents qui ont obtenu les meilleurs résultats et d'obtenir des effets de rétroaction. Le type d'information qu'ils peuvent modifier est choisi au hasard, il peut s'agir du nombre de voisins, de la liste des voisins les plus proches, de l'algorithme de récupération, de l'algorithme de réutilisation ou même des vecteurs bayésiens. Si un agent décide de modifier ses informations avec un voisin, les informations qu'il possédait à l'origine sont remplacées par une copie des informations obtenues auprès de l'agent sélectionné.

8.3/ RÉSULTATS

Afin de comparer la prédiction des performances et le comportement de l'algorithme proposé, onze bases de données de régression présentant des caractéristiques différentes ont été sélectionnées. Les bases de données et leurs caractéristiques sont indiquées dans le tableau 8.3. Les valeurs des paramètres de l'algorithme sont les suivantes

$it = 100$, $np = 50$, $nl = 10$ et $ng = 10$. Les paramètres de tous les autres algorithmes sont présentés dans le tableau 8.4.

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
DS1	9.081	12.301	1.228	1.066	7.763	9.081	9.078	9.764	0.750	8.225
DS2	0.022	0.025	0.019	0.013	0.017	0.022	0.022	0.037	0.011	0.016
DS3	8.756	8.465	9.656	7.665	8.716	8.756	9.005	9.177	7.369	7.991
DS4	0.647	0.752	0.794	0.602	0.688	0.647	0.646	0.798	0.616	0.607
DS5	0.767	0.824	0.877	0.66	0.826	0.767	0.775	0.87	0.703	0.662
DS6	10.525	9.174	6.93	5.372	6.662	10.525	10.525	10.527	5.131	9.070
DS7	2.961	2.451	0.589	0.528	3.955	2.961	3.009	4.083	0.490	2.941
DS8	1.298	1.125	1.360	1.197	1.486	1.298	1.298	1.306	1.128	2.553
DS9	2.256	2.565	3.174	2.377	2.817	2.256	2.255	2.468	2.293	2.468
DS10	3.136	3.415	4.173	3.165	3.710	3.136	3.135	3.161	3.108	3.621
DS11	0.625	0.565	0.741	0.56	0.606	0.626	0.626	0.681	0.541	0.54
Avg. Rank	6.4	6.9	8.2	2.6	7.2	6.45	6.35	9.55	2.1	4.75

TABLE 8.2 – Résultat de la métrique RMSE (Root Mean Squared Error) pour les bases de données évaluées avec des algorithmes de régression

ID	DataSet	Features	Instances	Output. Dimension	Input. Domain	Output Domain
DS1	Yatch Hydrodynamics	6	308	1	$\mathbb{R}$	$\mathbb{R}$
DS2	Electrical Grid Stability	12	10000	1	$\mathbb{R}$	$\mathbb{R}$
DS3	Real State Valuation	6	414	1	$\mathbb{R}_+$	$\mathbb{R}_+$
DS4	Wine Quality (Red)	11	1598	1	$\mathbb{R}_+$	$\mathbb{N}$
DS5	Wine Quality (White)	11	4897	1	$\mathbb{R}_+$	$\mathbb{N}$
DS6	Concrete Compressive Strength	8	1030	1	$\mathbb{R}_+$	$\mathbb{R}_+$
DS7	Energy Efficiency	8	768	2	$\mathbb{R}_+$	$\mathbb{R}_+$
DS8	Gas Turbine CO, NOx Emission (2015)	9	7384	2	$\mathbb{R}_+$	$\mathbb{R}_+$
DS9	Student Performace Portuguese	30	649	3	$\mathbb{N}*$	$\mathbb{N}$
DS10	Student Performance Math	30	395	3	$\mathbb{N}*$	$\mathbb{N}$
DS11	Generated Student Performance	5	1000	1	$\mathbb{R}_+$	$\mathbb{R}_+$

TABLE 8.3 – Description des bases de données évaluées. (* après codification comme *String*)

Le classement moyen de toutes les bases de données en fonction de la mesure RMSE (Root Mean Square Error) est indiqué dans le tableau 8.2. Les résultats détaillés des mêmes algorithmes et des mêmes bases de données, mais comparés avec la métrique MAE (Median Absolute Error), sont présentés dans le tableau 8.5.

La dispersion globale, la médiane et les valeurs aberrantes pour toutes les bases données évaluées sont présentées dans la figure 8.5, où l'on peut voir que l'algorithme proposé génère dans certains cas plus de valeurs aberrantes que d'autres algorithmes, mais la variance est très faible et la convergence est proche de la valeur réelle, meilleure que la plupart des algorithmes comparés. Les valeurs aberrantes sont plus élevées que les valeurs réelles.

Les résultats montrent que l'algorithme est compétitif dans la plupart des bases de données sélectionnées, obtenant les meilleurs résultats dans les bases DS2, DS4, DS5 et DS9. Globalement, d'après le classement moyen, l'algorithme est placé en troisième position, à proximité d'autres algorithmes d'ensemble. Par rapport à l'ESCBR, une amélioration des résultats et une réduction de la variance ont été obtenues, ce qui montre que les systèmes multi-agents et le raisonnement stochastique bayésien contribuent à l'apprentissage et à la convergence de l'algorithme vers des solutions plus proches de la solution réelle.

ID	Parameter	Value	ID	Parameter	Value
A1	Intercept	True	A6	Degree	4
	Positive	True		Bias	True
A2	Neighbors	5	A7	Fit Intercept	True
	Weights	Uniform		alpha	0.2
	Metric	Minkowsky		tol	1e-4
	Power Minkowsky	2			
A3	Error	Squared Error	A8	Fit Intercept	True
	Min samples split	2		alpha	[0.00001, 0.4]
				Max iter	1000
				tol	1e-4
A4	Estimators	10	A9	Error	Squarred Error
	Error	Squared Error		Learning Rate	0.1
	Min samples split	2		Estimators	100
	Bootstrap	True		Min Split	2
A5	Hidden Layers	100			
	Activation	Relu			
	Solver	Adam			
	alpha	0.0001			
	Learning Rate	0.001			
	Max Iter	200			
	beta1	0.9			
	beta2	0.999			
	epsilon	1e-8			

TABLE 8.4 – Paramètres pour tous les algorithmes comparés

Dataset	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
DS1	6.776	2.385	0.231	0.207	3.632	6.778	6.307	5.186	0.162	1.218
DS2	0.015	0.017	0.012	0.008	0.012	0.015	0.015	0.030	0.007	0.010
DS3	5.092	4.320	4.1	3.632	4.435	5.092	5.20	5.132	3.504	3.771
DS4	0.413	0.495	0.18	0.325	0.451	0.413	0.412	0.544	0.387	0.135
DS5	0.509	0.548	0.285	0.374	0.550	0.509	0.509	0.633	0.456	0.085
DS6	6.989	5.709	3.134	2.839	4.306	6.989	6.989	6.986	3.084	5.072
DS7	1.393	1.372	0.217	0.218	2.523	1.393	1.529	2.346	0.243	1.006
DS8	0.549	0.297	0.365	0.289	0.742	0.549	0.549	0.540	0.309	0.794
DS9	1.496	1.788	2.080	1.612	2.005	1.496	1.496	1.714	1.538	1.556
DS10	2.344	2.534	2.910	2.331	2.543	2.344	2.344	2.481	2.258	2.371
DS11	0.387	0.35	0.46	0.338	0.384	0.387	0.387	0.453	0.327	0.347
Avg. Rank	7.15	6.9	5.35	2.6	7.95	7.25	7.3	9.0	2.5	4.5

TABLE 8.5 – Comparaison des résultats MAE (Median Absolute Error) pour les bases de données évaluées avec des algorithmes de régression

8.4/ CONCLUSION

L'ECBR proposé avec SMA tente d'utiliser les avantages des systèmes multi-agents pour améliorer la qualité des solutions proposées ainsi que pour améliorer le processus d'apprentissage global afin d'augmenter la performance de l'algorithme à chaque nouvelle exécution même sur des données différentes, il utilise également le raisonnement bayésien qui permet d'obtenir de bonnes approximations et un apprentissage optimal avec

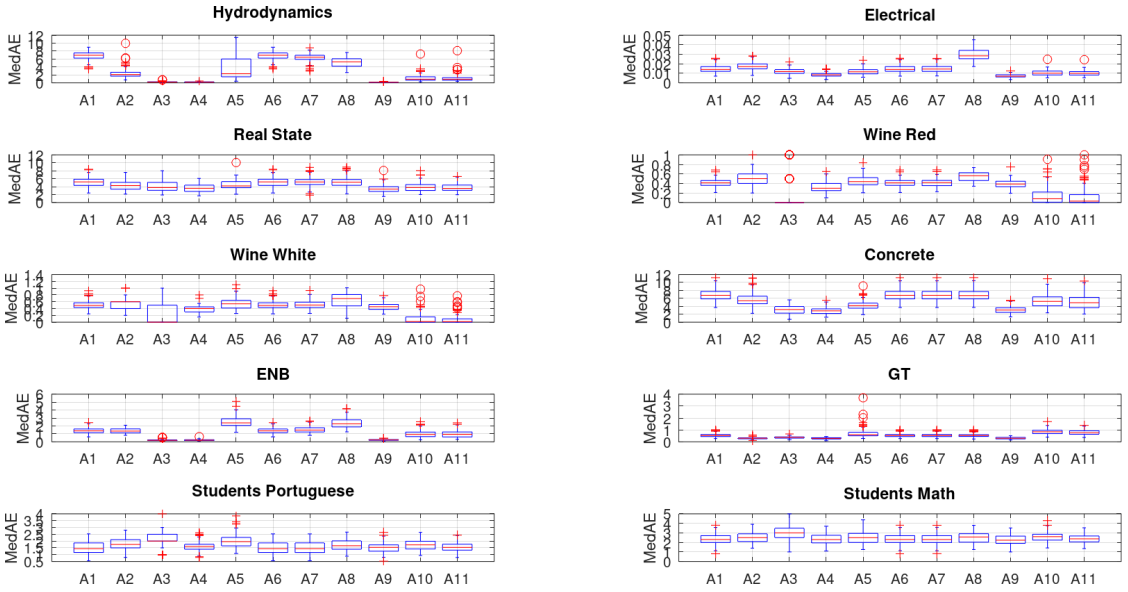

FIGURE 8.5 – Résultats de la métrique MAE (Median Absolute Error) pour dix algorithmes

peu de données.

Ce travail a démontré la capacité du modèle à trouver des solutions proches de l'optimum global pour la majorité des ensembles de données analysés, même avec des ensembles de données divers, déséquilibrés et de tailles différentes. Les résultats montrent en effet une amélioration de l'ECBR de base, qui peut être due à la combinaison du potentiel de l'ECBR et des caractéristiques des systèmes multi-agents tels que les effets de rétroaction, les effets émergents et le comportement de raisonnement bayésien cognitif mis en œuvre.

9

APPLICATION DU RAISONNEMENT À PARTIR DE CAS (RÀPC) ET DES SYSTÈMES MULTI-AGENT (SMA) AU SYSTÈME AI-VT

9.1/ INTRODUCTION

Ce chapitre présente l'intégration de tous les algorithmes développés et explicités dans les chapitres précedents. Le modèle integré est appliqué au AI-VT système sur une base de données générée et une base de données réelle. Plusieurs types de test sont executés pour montrer que le modèle final permet en effet d'amméliorer les capacités d'identification et adaptation.

Les contributions de ce chapitre sont les suivantes :

— Vérification de l'efficacité du modèle de raisonnement basé sur les cas pour la prédiction avec une base de données de notes d'apprenants par rapport à d'autres algorithmes.
— Calcul explicite de l'évolution de l'acquisition des connaissances en analysant le changement des distributions de probabilité générées par le modèle de recommandation stochastique.
— Intégration du modèle de recommandation stochastique à la prédiction par raisonnement basé sur les cas pour améliorer la personnalisation de l'ITS.

9.2/ CONCEPTS ASSOCIÉS

Cette section présente les concepts, les définitions et les algorithmes nécessaires à la compréhension du modèle proposé, ainsi que les modèles et les mesures fondamentaux. Le premier paradigme fondamental utilisé dans ce travail est le raisonnement à partir de cas (CBR), qui permet d'exploiter les connaissances historiquement acquises et l'expérience accumulée en ce qui concerne un problème spécifique. Ce paradigme est utilisé pour générer des solutions émergentes pour un nouveau problème en utilisant une base de données de connaissances. L'idée principale est de rechercher des situations antérieures similaires et d'utiliser l'expérience acquise pour résoudre de nouveaux

problèmes. La CBR est particulièrement utile lorsque les causes sous-jacentes d'un problème ne sont pas bien comprises. Le raisonnement à base de cas définit un cycle de quatre étapes pour améliorer la solution d'inférence [**?**].

Puisque l'objectif ici est d'adapter les exercices proposés par AI-VT, il est nécessaire de connaître le fonctionnement de l'un des algorithmes les plus utilisés pour effectuer l'adaptation du contenu et des exercices dans certains STI, afin de comparer les résultats avec l'algorithme proposé et de voir dans quelle mesure il permet d'obtenir une amélioration de l'adaptation et de la performance des apprenants. L'un des modèles les plus couramment utilisés dans les STI pour adapter le contenu et estimer la progression du niveau de connaissance des apprenants est le BKT (Bayesian Knowledge Tracing) [**?**]. Ce modèle utilise quatre paramètres pour estimer la progression des connaissances. $P(k)$ estime la probabilité de connaissance dans une compétence spécifique. $P(w)$, est la probabilité que l'apprenant démontre ses connaissances. $P(s)$, est la probabilité que l'apprenant fasse une erreur.$P(g)$, est la probabilité que l'apprenant ait deviné une réponse. La valeur estimée de la connaissance est mise à jour avec les équations 9.1, 9.2 et 9.3. Si la réponse de l'apprenant est correcte, l'équation 9.1 est utilisée, mais si la réponse est incorrecte, l'équation 9.2 est utilisée.

$$P(k_{t-1}|Correct_t) = \frac{P(k_{t-1})(1-P(s))}{P(k_{t-1})(1-P(s)) + (1-P(k_{t-1}))P(g)} \tag{9.1}$$

$$P(k_{t-1}|Incorrect_t) = \frac{P(k_{t-1})P(s)}{P(k_{t-1})(P(s)) + (1-P(k_{t-1}))(1-P(g))} \tag{9.2}$$

$$P(k_t) = P(k_{t-1}|evidence_t) + (1-P(k_{t-1}|evidence_t))P(w) \tag{9.3}$$

Le modèle de recommandation proposé, associé à AI-VT, est basé sur le paradigme de l'apprentissage par renforcement. L'apprentissage par renforcement est une technique d'apprentissage automatique qui permet, par le biais d'actions et de récompenses, d'améliorer les connaissances du système sur une tâche spécifique [**?**]. L'algorithme utilisé pour l'adaptation est un algorithme d'apprentissage par renforcement appelé échantillonnage de Thompson, qui, par le biais d'une distribution de probabilité initiale (distribution a priori) et d'un ensemble de règles de mise à jour prédéfinies, peut adapter et améliorer les estimations initiales d'un processus analysé spécifique [**?**]. La distribution de probabilité initiale est généralement définie comme une distribution spécifique de la famille des distributions Bêta (équation 9.4) avec des valeurs initiales prédéterminées pour α et β [**?**], [**?**].

$$Beta(\theta|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1} \tag{9.4}$$

En utilisant la definition formelle de la fonction Gamma Γ (équation 9.5) et en remplaçant des variables, une nouvelle expression de la fonction Beta est obtenue (équation 9.6).

$$\Gamma(z) = \int_0^\infty e^{-x}x^{z-1}dx \tag{9.5}$$

$$Beta(\theta|\alpha,\beta) = \frac{\int_0^\infty e^{-s}s^{\alpha+\beta-1}ds}{\int_0^\infty e^{-u}u^{\alpha-1}du \int_0^\infty e^{-v}v^{\beta-1}dv}\theta^{\alpha-1}(1-\theta)^{\beta-1} \tag{9.6}$$

En exprimant les deux intégrales du denominateur comme une seule intégrale, l'équation 9.7 est obtenue.

$$\int_{u=0}^{\infty}\int_{v=0}^{\infty} e^{-u-v}u^{\alpha-1}v^{\beta-1}dudv \tag{9.7}$$

Après, sont remplacées $u = st$, $v = s(1-t)$, $s = u+v$ et $t = u/(u+v)$, avec le résultat du Jacobien 9.8, alors l'expression finale est comme montre l'équation 9.9.

$$\begin{pmatrix} \frac{\partial u}{\partial t} & \frac{\partial u}{\partial s} \\ \frac{\partial v}{\partial t} & \frac{\partial v}{\partial s} \end{pmatrix} = \begin{pmatrix} sdt & tds \\ -sdt & (1-t)ds \end{pmatrix} = s\ dtds \tag{9.8}$$

$$\int_{s=0}^{\infty}\int_{t=0}^{1} e^{-s}(st)^{\alpha-1}(s(1-t))^{\beta-1}s\ dsdt \tag{9.9}$$

Si les intégrales sont exprimées en fonction des variables indépendantes s et t l'équation 9.10 est générée.

$$\int_{s=0}^{\infty} e^{-s}s^{\alpha+\beta-1}ds \int_{t=0}^{1} t^{\alpha-1}(1-t)^{\beta-1}dt \tag{9.10}$$

En plaçant les termes dans l'équation le résultat est l'équation 9.11.

$$Beta(\theta|\alpha,\beta) = \frac{\int_{0}^{\infty} e^{-s}s^{\alpha+\beta-1}ds}{\int_{s=0}^{\infty} e^{-s}s^{\alpha+\beta-1}ds \int_{t=0}^{1} t^{\alpha-1}(1-t)^{\beta-1}dt}\theta^{\alpha-1}(1-\theta)^{\beta-1} \tag{9.11}$$

Finalement, la famille de fonctions de distribution Beta peut être exprimée comme l'équation 9.12. Les métriques utilisées dans ce chapitre s'expriment en fonction de cette définition.

$$Beta(\theta|\alpha,\beta) = \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{\int_{0}^{1} t^{\alpha-1}(1-t)^{\beta-1}dt} \tag{9.12}$$

L'évolution de l'algorithme de recommandation TS est établie par le changement des distributions de probabilité, mais au moment de quantifier l'évolution, le changement et la variabilité doivent être calculés en fonction du temps. Les distributions de probabilités peuvent être comparées pour déterminer leur degré de similitude, sous la forme d'une métrique qui détermine numériquement les différences entre elles. L'apprentissage automatique utilise la divergence de Kullback-Liebler, qui décrit l'entropie relative de deux distributions de probabilités. Cette fonction est basée sur le concept d'entropie et le résultat peut être interprété comme la quantité d'information nécessaire pour obtenir la distribution de probabilité q à partir de la distribution de probabilité p. La divergence de Kullback-Liebler (équation 9.13) est largement utilisée, mais elle présente l'inconvénient de ne pas pouvoir être utilisée comme métrique dans certains cas, car il ne s'agit pas d'une mesure symétrique, $D_{KL}(p,q) \neq D_{KL}(q,p)$, elle ne satisfait pas à l'inégalité triangulaire et elle n'est pas bornée [**?**]. Pour remédier à cette difficulté, il est possible d'utiliser la divergence de Jensen-Shannon.

$$D_{KL}(p(x), q(x)) = \int_{-\infty}^{\infty} p(x) log\left(\frac{p(x)}{q(x)}\right) dx \tag{9.13}$$

La divergence de Jenser-Shannon est basée sur la divergence de Kullback-Liebler, à la différence qu'une distribution de probabilité auxiliaire m est créée dont la définition est basée sur les distributions initiales p et q [**?**]. L'équation 9.14 montre la définition formelle de la divergence de Jensen-Shannon, où $m(x)$ est une distribution de mélange de probabilités basée sur $p(x)$ et $q(x)$, l'équation 9.15 montre comment elle est calculée. La divergence de Jensen-Shannon est un mélange de distributions de probabilités basé sur $p(x)$ et $q(x)$.

$$D_{JS}(p(x), q(x)) = \frac{1}{2} D_{KL}(p(x), m(x)) + \frac{1}{2} D_{KL}(q(x), m(x)) \tag{9.14}$$

$$m(x) = \frac{1}{2} p(x) + \frac{1}{2} q(x) \tag{9.15}$$

Les distributions de probabilité à comparer doivent être continues et définies dans le même domaine.

La prédiction utilisée dans le modèle proposé est basée sur les travaux de Soto *et al.* [**?**], il s'agit d'un modèle d'empilage de raisonnement basé sur les cas qui met en œuvre deux niveaux d'intégration, le modèle utilise globalement la stratégie d'empilage pour exécuter plusieurs algorithmes afin de rechercher des informations dans un ensemble de données et de générer des solutions à différents problèmes génériques, en outre il y a une étape d'évaluation qui permet de sélectionner la solution la plus optimale pour un problème donné en fonction d'une métrique adaptative définie pour les problèmes de régression. Il a été décidé de mettre en œuvre le modèle basé sur l'empilement car il s'agit d'une méthode d'ensemble qui permet d'éviter le paradoxe de Stein puisqu'elle combine les points de vue de différents estimateurs à des étapes de récupération et de réutilisation par raisonnement basé sur les cas.

9.3/ MODÈLE PROPOSÉ

Le modèle proposé est une intégration du modèle d'adaptation stochastique (basé sur l'échantillonnage de Thompson) avec le raisonnement à base de cas d'ensemble (ESCBR-SMA). Dans ce cas, le modèle de recommandation produit une adaptation en fonction des notes de l'apprenant et l'ESCBR-SMA effectue une prédiction pour valider l'adaptation générée.

L'idée d'unifier les deux modèles est d'obtenir des informations du point de vue local où une recommandation est obtenue en se basant uniquement sur les informations des apprenants individuels (modèle basé sur l'échantillonnage de Thompson) et la prédiction globale où les informations sont obtenues à partir de tous les apprenants qui ont des résultats similaires (filtre collaboratif avec CBR). L'architecture du modèle est présentée dans la figure 10.1, où l'on peut voir que les deux modèles TS et CBR sont exécutés en parallèle et indépendamment avec les informations extraites de la même base de données, une fois que les résultats de chaque modèle sont obtenus, les résultats sont unifiés

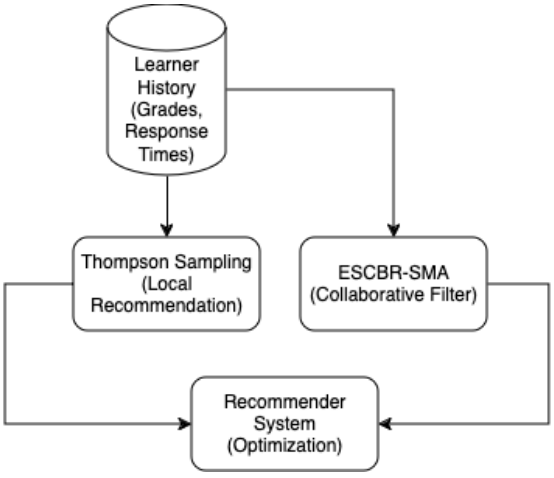

FIGURE 9.1 – Schéma de l'architecture du modèle proposé

par le biais d'une fonction de pondération, la recommandation finale est celle qui maximise l'expression 10.3. La consolidation des résultats des deux modèles permet d'atténuer l'effet du paradoxe de Simpson [**?**]. Ce paradox décrit l'effet qui se présente lorsque les données sont grouppes de différents manières et montrent tendances divergentes [**?**].

La première étape est l'adaptation avec l'échantillonnage de Thompson, puis la prédiction ECBR-SMA et enfin la prise de décision à envoyer à l'apprenant. Le système de recommandation obtient une valeur de probabilité pour tous les niveaux de complexité de l'apprenant et l'ECBR-SMA évalue la proposition avec une prédiction pour chaque niveau de complexité. Le tableau 10.1 présente les variables et les paramètres du modèle proposé ainsi que les mesures employées. Le tableau 10.1 présente les variables et les paramètres du modèle proposé ainsi que les mesures employées.

ID	Type	Description	Domain
α	p	Paramètre de la distribution bêta	$[1, \infty] \in \mathbb{R}$
β	p	Paramètre de la distribution bêta	$[1, \infty] \in \mathbb{R}$
t	p	Temps défini comme itérations	$\mathbb{N}$
c	p	Niveau de complexité	$\mathbb{N}$
$k_{t,c}$	v	Évolution de la connaissance dans le temps t pour le niveau de complexité c	$[0, 1] \in \mathbb{R}$
$vk_{t,c}$	v	Évolution de la connaissance pour chaque niveau de complexité c	$\mathbb{R}$
TS_c	v	Récompense d'échantillonnage de Thompson pour un niveau de complexité c	$[0, 1] \in \mathbb{R}$
TSN_c	v	Normalization de TS_c avec d'autres niveaux de complexité	$[0, 1] \in \mathbb{R}$
$ESCBR_c$	v	Prédiction de la note pour un niveau de complexité c	$\mathbb{R}_+$
p_c	f	Fonction de densité de probabilité pour le niveau de complexité c	$\mathbb{R}_+$
D_{JS}	f	Divergence de Jensen-Shannon	$[0, 1] \in \mathbb{R}$

TABLE 9.1 – Paramètres (p), variables (v) et fonctions (f) du modèle proposé et les métriques utilisées

L'intégration se fait en trois étapes. Tout d'abord, il est nécessaire d'avoir des valeurs aléatoires pour chaque niveau de complexité c en utilisant les distributions de probabilité générées avec le modèle TS (équation 10.1), une fois que toutes les valeurs de proba-

bilité correspondant à tous les niveaux de complexité ont été obtenues, la normalisation de toutes ces valeurs est calculée comme indiqué dans l'équation 10.2. Les valeurs de normalisation servent de paramètres de priorité pour les prédictions effectuées par le modèle ESCBR-SMA, comme le montre l'équation 10.3.

$$TS_c = rand(Beta(\alpha_c, \beta_c)) \tag{9.16}$$

$$TSN_c = \frac{TS_c}{\sum_{i=0}^{4} TS_i} \tag{9.17}$$

$$n_c = argmax_c(TSN_c * ESCBR_c) \tag{9.18}$$

Avec les valeurs finales calculées pour chaque niveau de complexité, le niveau de complexité qui a la valeur la plus élevée est proposé comme recommandation finale (équation 10.3). Le niveau de complexité qui a la valeur la plus élevée est proposé comme recommandation finale (équation 10.3).

9.4/ RÉSULTATS ET DISCUSSION

Cette section présente la description de la base de données et les paramètres utilisés pour mesurer la précision, la performance et la progression des connaissances, les résultats individuels du modèle de recommandation, le modèle de prédiction ainsi que leur intégration finale pour améliorer la personnalisation du système d'IA-VT. Cette section présente les résultats individuels du modèle de recommandation, le modèle de prédiction ainsi que leur intégration finale pour améliorer la personnalisation du système d'IA-VT.

La base de données a été générée avec la distribution logit-normale pour simuler les notes des apprenants, car il s'agit d'un bon modèle pour se rapprocher du monde réel. La base de données représente les notes et les temps de réponse d'un apprenant pour cinq questions à chaque niveau de complexité.

Le principal inconvénient de ce système de validation « en situation réelle » est la difficulté de la collecte des données. Cette difficulté est accentuée dans les contextes d'apprentissage autorégulé, puisque les apprenants peuvent quitter la plateforme d'apprentissage à tout moment et que les données peuvent être incomplètes [**?**].

Quatre tests différents ont été effectués pour démontrer les avantages de l'intégration de la TS et de la CBR dans les EIAH. Le premier est l'utilisation de CBR pour la régression avec une base de données d'apprenants afin de démontrer la capacité du modèle à prédire les notes à différents niveaux de complexité, le deuxième est l'évaluation de la progression des connaissances avec TS afin de déterminer l'efficacité du modèle dans la recommandation personnalisée pour chaque apprenant, La troisième est la comparaison entre les modèles de recommandation BKT et TS afin d'établir la performance du modèle TS en utilisant BKT comme modèle de base et enfin, la comparaison entre TS seul et TS avec ESCBR-SMA pour démontrer que l'intégration entre les deux modèles améliore l'ensemble du système de recommandation dans AI-VT.

9.4.1/ RÉGRESSION DANS LA BASE DE DONNÉES DES APPRENANTS AVEC ESCBR-SMA

Le SMA utilise le raisonnement bayésien, ce qui permet aux agents d'apprendre des données et des interactions au cours de l'exécution et de l'exploration.

L'algorithme utilise une fonction noyau pour obtenir la meilleure approximation de la solution du nouveau problème, le problème de l'obtention de la meilleure solution est un problème NP, car la formulation est similaire au problème de Fermat-Weber à N dimensions. Le problème de l'obtention de la meilleure solution est un problème NP, car la formulation est similaire au problème de Fermat-Weber à N dimensions [**?**].

La première série de tests est définie sous la forme de différents scénarios, comme le montre le tableau 9.2. Dans le scénario 1 (E1), il s'agit de prédire la note d'un apprenant au premier niveau de complexité, après 3 questions. Le scénario 2 (E2) contient les notes de 8 questions et l'objectif est de prédire la note de 9 questions dans le même niveau de complexité. Le scénario 3 (E3) contient les données permettant de prédire le passage à un niveau de complexité supérieur après 4 questions. Le scénario 4 (E4) contient 4 questions et la prédiction de 2 notes dans un niveau de complexité supérieur.

Scenario	Features	Output Dimension
E1	5	1
E2	15	1
E3	9	1
E4	9	2

TABLE 9.2 – Description des scénarios

Le modèle a été comparé à neuf algorithmes bien connus utilisés pour résoudre les problèmes de régression. La liste des algorithmes est présentée dans le tableau 9.3.

ID	Algorithm	ID	Algorithm
A1	Linear Regression	A6	Polinomial Regression
A2	K-Nearest Neighbor	A7	Ridge Regression
A3	Decision Tree	A8	Lasso Regression
A4	Random Forest (Ensemble)	A9	Gradient Boosting (Ensemble)
A5	Multi Layer Perceptron	A10	Proposed Ensemble Stacking CBR

TABLE 9.3 – Liste des algorithmes évalués

Les algorithmes ont été évalués à l'aide de trois mesures (Root Mean Squared Error - RMSE, Median Absolute Error - MedAE, Mean Absolute Error - MAE), dont les résultats figurent dans le tableau 9.4, où l'on constate que l'algorithme proposé obtient de meilleurs résultats que les autres algorithmes avec lesquels il a été comparé, sauf dans les cas E1(MedAE), E1(MAE), E2(MedAE), E2(MAE), E3 et E4(MedAE) où les meilleurs résultats sont obtenus par l'algorithme A9, mais l'algorithme proposé occupe la deuxième place dans ces cas avec des résultats très proches. Il est possible de conclure que l'intégration de plusieurs algorithmes de recherche et de génération de solutions dans le cadre des paradigmes CBR et Stacking est efficace dans le cas de l'application à la prédiction des notes des apprenants.

	Algorithme									
Scenario (Metrique)	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
E1 (RMSE)	0.625	0.565	0.741	0.56	0.606	0.626	0.626	0.681	0.541	**0.54**
E1 (MedAE)	0.387	0.35	0.46	0.338	0.384	0.387	0.387	0.453	**0.327**	0.347
E1 (MAE)	0.485	0.436	0.572	0.429	0.47	0.485	0.485	0.544	**0.414**	0.417
E2 (RMSE)	0.562	0.588	0.78	0.571	0.61	0.562	0.562	0.622	0.557	**0.556**
E2 (MedAE)	0.351	0.357	0.464	0.344	0.398	0.351	0.351	0.415	**0.334**	0.346
E2 (MAE)	0.433	0.448	0.591	0.437	0.478	0.433	0.433	0.495	**0.422**	0.429
E3 (RMSE)	0.591	0.59	0.79	0.57	0.632	0.591	0.591	0.644	**0.555**	0.558
E3 (MedAE)	0.367	0.362	0.474	0.358	0.404	0.367	0.367	0.433	**0.336**	0.349
E3 (MAE)	0.453	0.45	0.598	0.441	0.49	0.453	0.453	0.512	**0.427**	0.43
E4 (RMSE)	0.591	0.589	0.785	0.568	0.613	0.591	0.591	0.644	0.554	**0.549**
E4 (MedAE)	0.367	0.362	0.465	0.57	0.375	0.367	0.367	0.433	**0.336**	0.343
E4 (MAE)	0.453	0.45	0.598	0.438	0.466	0.453	0.453	0.512	0.426	**0.417**

TABLE 9.4 – Résultats de la régression pour la base de données des apprenants avec 100 exécutions

9.4.2/ PROGRESSION DES CONNAISSANCES

Le modèle de recommandation TS est basé sur le paradigme bayésien, ce qui est très utile lorsque les données sont limitées et l'incertitude forte. Afin de quantifier la connaissance et de voir sa progression dans le temps avec TS, la divergence de Jensen-Shannon avec la famille de distribution Beta en t et $t-1$ fois a été utilisée comme second test. L'équation 9.19 décrit formellement le calcul à effectuer avec les distributions de probabilité en un temps t pour un niveau de complexité c, en utilisant la définition m (équation 9.20).

$$k_{t,c} = \frac{1}{2}\int_0^1 p_c(\alpha_t,\beta_t,x) log\left(\frac{p_c(\alpha_t,\beta_t,x)}{m(p_c(\alpha_{t-1},\beta_{t-1},x),p_c(\alpha_t,\beta_t,x))}\right)dx + \frac{1}{2}\int_0^1 p_c(\alpha_{t-1},\beta_{t-1},x) log\left(\frac{p_c(\alpha_{t-1},\beta_{t-1},x)}{m(p_c(\alpha_{t-1},\beta_{t-1},x),p_c(\alpha_t,\beta_t,x))}\right)dx \quad (9.19)$$

$$m(p(\alpha_{(t-1)},\beta_{(t-1)},x),p(\alpha_t,\beta_t,x)) = \frac{1}{2}\left(\frac{x^{\alpha_{(t-1)}-1}(1-x)^{\beta_{(t-1)}-1}}{\int_0^1 u^{\alpha_{(t-1)}-1}(1-u^{\beta_{(t-1)}-1})du}\right) + \frac{1}{2}\left(\frac{x^{\alpha_t-1}(1-x)^{\beta_t-1}}{\int_0^1 u^{\alpha_t-1}(1-u^{\beta_t-1})du}\right) \quad (9.20)$$

La progression totale des connaissances en t est la somme des différences entre t et $t-1$ pour tous les c niveaux de complexité calculés avec la divergence de Jensen-Shannon (équation 9.22). en utilisant l'évaluation de la progression de la variabilité (équation 9.21).

$$vk_{t,c} = \begin{cases} D_{JS}(Beta(\alpha_{t,c},\beta_{t,c}),Beta(\alpha_{t+1,c},\beta_{t+1,c})), & \frac{\alpha_{t,c}}{\alpha_{t,c}+\beta_{t,c}} < \frac{\alpha_{t+1,c}}{\alpha_{t+1,c}+\beta_{t+1,c}} \\ -D_{JS}(Beta(\alpha_{t,c},\beta_{t,c}),Beta(\alpha_{t+1,c},\beta_{t+1,c})), & Otherwise \end{cases} \quad (9.21)$$

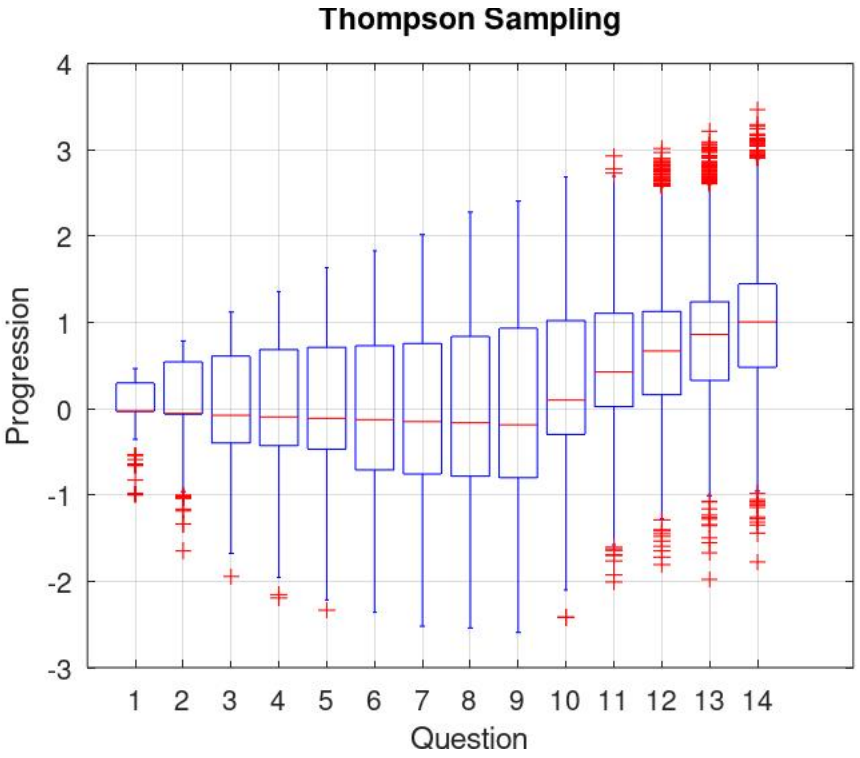

FIGURE 9.2 – Progression des connaissances avec l'échantillonnage de Thompson selon la divergence de Jensen-Shannon

$$k_t = \sum_{c=4}^{c=0 \vee k_t \neq 0} \begin{cases} \alpha_{c-1} vk_{t,c-1}; & vk_{t,c} > 0 \\ 0; & Otherwise \end{cases} \tag{9.22}$$

La figure 9.2 montre la progression cumulative des connaissances sur les quinze questions d'une seule session de formation. Entre la première et la dernière question de la même session, tous les apprenants ont statistiquement augmenté leur niveau de connaissance puisque la moyenne a augmenté, la variabilité augmente à partir de la première question jusqu'à la question neuf, où le système a acquis plus d'informations sur les apprenants, à partir de là la variabilité diminue et la moyenne augmente. La figure fig :evolution montre la progression cumulative des connaissances sur les quinze questions d'une même session de formation.

9.4.3/ COMPARAISON ENTRE TS ET BKT

L'évolution du système de recommandation TS est testée en comparaison avec BKT, la figure 9.3 montre l'évolution des notes des apprenants en fonction du nombre de questions auxquelles ils répondent dans la même session. Dans ce cas, le modèle TS génère moins de variabilité que BKT, mais si est faite la comparaison des moyennes générées par chaque question, l'évolution est très similaire. La figure 9.3 montre l'évolution des notes des apprenants en fonction du nombre de questions auxquelles ils répondent au cours de la même session.

Mais, si les résultats obtenus sont comparés par rapport à l'évolution du niveau de com-

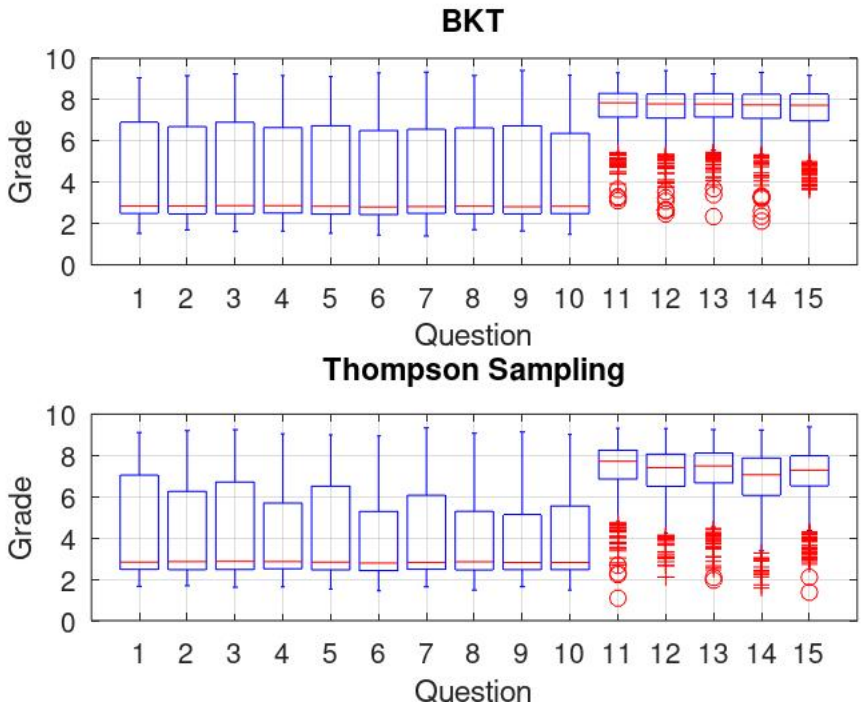

FIGURE 9.3 – Comparaison de l'évolution des notes entre les algorithmes BKT et TS

plexité recommandé (figure 9.4), le modèle TS fait évoluer le niveau de complexité des apprénants, alors que le modèle BKT a tendance à laisser les apprénants au même niveau de complexité, c'est-à-dire qu'avec le modèle BKT, il est difficile d'apprendre de nouveaux sujets ou des concepts plus complexes au sein du même domaine. En examinant les résultats des deux figures (figures 9.3 et 9.4) et en établissant des comparaisons, le modèle TS permet de progresser en moyenne dans la valeur des notes et facilite l'évolution des niveaux de complexité.ℕ- Le modèle TS permet de progresser en moyenne dans la valeur des notes et facilite l'évolution des niveaux de complexité. Le modèle TS permet de progresser en moyenne dans la valeur des notes et facilite l'évolution des niveaux de complexité.

9.4.4/ SYSTÈME DE RECOMMANDATION AVEC ESCBR-SMA

Le troisième test est l'intégration entre deux modèles. Cette combinaison est faite pour éviter le paradoxe de Stein, en essayant de combiner des observations qui ne sont pas directement liées l'une à l'autre, c'est-à-dire en utilisant l'information individuelle (Thomson sampling recommender) et le filtre collaboratif (Case-base reasoning prediction) pour améliorer la personnalisation. Le test est une comparaison entre le système de recommandation TS et le système de recommandation TS avec la prédiction ESCBR-SMA afin de déterminer si l'intégration des deux modèles permet d'améliorer l'évolution du processus d'apprentissage proposé par le système AI-VT.

La comparaison est effectuée après la question 6 pour tous les apprenants, car il est nécessaire de disposer d'informations préalables pour utiliser l'algorithme ESCBR-SMA et prédire les notes dans tous les niveaux de complexité pour la question suivante.

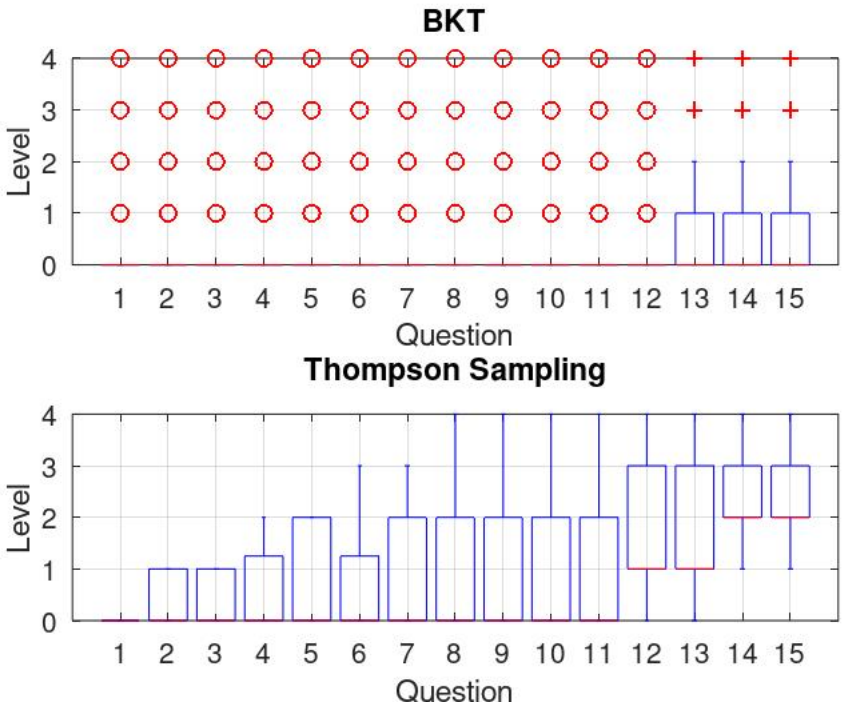

FIGURE 9.4 – Comparaison de l'évolution des niveaux entre les algorithmes BKT et TS

9.4.5/ PROGRESSION DES CONNAISSANCES TS VS TS ET ESCBR-SMA

Pour établir la différence entre le modèle de recommandation TS et le même modèle associé à la prédiction basée sur le raisonnement à partir de cas ESCBR-SMA, le quatrième test est défini en utilisant la métrique de Jensen-Shannon, mais dans ce cas la comparaison est faite entre les différents modèles (TS, TS-ESCBR) sur le même niveau de complexité dans le même temps t. La définition formelle de la métrique est exprimée par les équations 9.23 et 9.24. La définition formelle de la métrique est exprimée par les équations 9.23 et 9.24.

$$k_{t,c} = \frac{1}{2}\int_0^1 p_c(\alpha_{p1,t},\beta_{p1,t},x)log\left(\frac{p_c(\alpha_{p1,t},\beta_{p1,t},x)}{m(p_c(\alpha_{p1,t},\beta_{p1,t},x),p_c(\alpha_{p2,t},\beta_{p2,t},x))}\right)dx + \frac{1}{2}\int_0^1 p_c(\alpha_{p2,t},\beta_{p2,t},x)log\left(\frac{p_c(\alpha_{p2,t},\beta_{p2,t},x)}{m(p_c(\alpha_{p1,t},\beta_{p1,t},x),p_c(\alpha_{p2,t},\beta_{p2,t},x))}\right)dx \quad (9.23)$$

$$m(p(\alpha_{p1,t},\beta_{p1,t},x),p(\alpha_{p2,t},\beta_{p2,t},x)) = \frac{1}{2}\left(\frac{x^{\alpha_{p1,t}-1}(1-x)^{\beta_{p1,t}-1}}{\int_0^1 u^{\alpha_{p1,t}-1}(1-u^{\beta_{p1,t}-1})du}\right) + \frac{1}{2}\left(\frac{x^{\alpha_{p2,t}-1}(1-x)^{\beta_{p2,t}-1}}{\int_0^1 u^{\alpha_{p2,t}-1}(1-u^{\beta_{p2,t}-1})du}\right) \quad (9.24)$$

La comparaison entre l'évolution des connaissances présente une bifurcation après la septième question, et l'intégration de l'échantillonnage de Thompson et du raisonnement

à partir de cas permet d'améliorer l'évolution des connaissances par rapport au seul modèle d'échantillonnage de Thompson (Figure 9.5). Pour toutes les questions de la même session, en moyenne, la progression est supérieure par rapport à l'utilisation du seul modèle d'échantillonnage de Thompson comme modèle de recommandation.

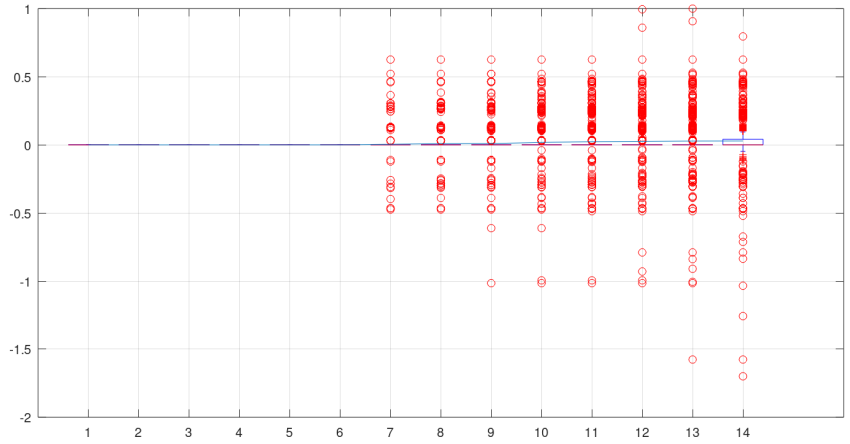

FIGURE 9.5 – Normalisation de la différence de progression entre l'échantillonnage de Thompson et l'échantillonnage de Thompson avec ESCBR pour 1000 apprenants

9.5/ CONCLUSION

Ce chapitre présente un modèle intégré entre deux modèles développés précédemment, un système de recommandation basé sur l'algorithme d'échantillonnage de Thompson et un modèle de régression d'ensemble basé sur le raisonnement par cas. Le modèle intégré est appliqué à un ITS appelé AI-VT, les résultats montrent en effet que l'intégration permet d'améliorer la performance des deux modèles utilisés séparément, en outre il montre de meilleurs résultats dans la révision/adaptation des étapes de solutions pour chaque apprenant, en fonction des métriques utilisées et des tests définis, donnant une meilleure personnalisation du système et facilitant l'acquisition de connaissances.

Les avantages du modèle proposé sont les suivants : i) il permet de générer des recommandations personnalisées pour chaque apprenant avec relativement peu de données historiques, ii) étant donné que de multiples points de vue (différents algorithmes) sur le même problème et avec la même base de données sont intégrés, le risque de tomber dans des paradoxes statistiques (Stein, Simpson) est réduit, iii) les deux modèles se complètent mutuellement en améliorant les résultats finaux d'une manière généralisée. Le modèle proposé a été conçu pour être utilisé dans le cadre d'un projet de recherche et de développement en cours.

10

APPLICATION DU RAISONNEMENT À PARTIR DE CAS (RÀPC), DES SYSTÈMES MULTI-AGENTS (SMA) ET DU PROCESSUS DE HAWKES AU SYSTÈME AI-VT

10.1/ INTRODUCTION

L'un des principaux modules des EIAH est le système de recommandation, qui vise à trouver les faiblesses et à adapter la plateforme localement ou globalement pour faciliter le processus d'apprentissage et l'acquisition des connaissances, ce module est très important car il permet d'adapter le système et de personnaliser les contenus et les exercices en fonction des besoins et des résultats de chacun des apprenants, l'efficacité du système dans l'acquisition des connaissances et l'adaptation aux différents types d'apprentissage dépend de ce module [**?**]. Il est donc nécessaire de trouver des techniques et des algorithmes capables d'exploiter les données disponibles et d'explorer les options d'apprentissage de manière dynamique, afin d'améliorer les performances globales des EIAH.

Dans ce chapitre seront detaillées les contributions réalisées avec l'incorporation du processus de Hawkes :

- Simulation de la courbe d'oubli dans le processus d'apprentissage à l'aide du processus stochastique de Hawkes.
- Intégration du raisonnement par cas, des systèmes multi-agents et du processus de Hawkes dans un algorithme de recommandation.
- Vérification de la progression, de la stabilité, la précision et évolution de l'algorithme de recommandation stochastique proposé à l'aide de bases de données simulées et hétérogènes d'étudiants réels.

10.2/ MODÈLE PROPOSÉ

L'algorithme proposé est une intégration de la recommandation stochastique (basée sur l'échantillonnage de Thompson), du raisonnement à partir de cas (ESCBR-SMA) et du processus de Hawkes. Dans ce cas, l'algorithme de recommandation produit une adaptation en fonction des notes de l'apprenant et l'ESCBR-SMA effectue une prédiction pour valider l'adaptation générée, le processus de Hawkes simule la courbe d'oubli dans le processus d'apprentissage.

L'idée de l'unification est d'obtenir des informations d'un point de vue local où une recommandation est obtenue en se basant uniquement sur les informations des apprenants individuels (modèle basé sur l'échantillonnage de Thompson), la prédiction globale où les informations sont obtenues à partir de tous les apprenants qui ont des résultats similaires (filtre collaboratif avec CBR) et le processus d'apprentissage dynamique avec le processus de Hawkes. L'architecture de l'algorithme est présentée dans la figure 10.1, où TS et CBR sont exécutés en parallèle et indépendamment avec les informations extraites de la même base de données, une fois que les résultats de chaque algorithme sont obtenus, les résultats sont unifiés par une fonction de pondération et une distribution de probabilité mises à jour dynamiquement en fonction des événements passés et du niveau de complexité sélectionné, la recommandation finale est celle qui maximise l'expression 10.3. La consolidation des deux résultats globaux permet d'atténuer l'effet du paradoxe de Simpson [**?**].

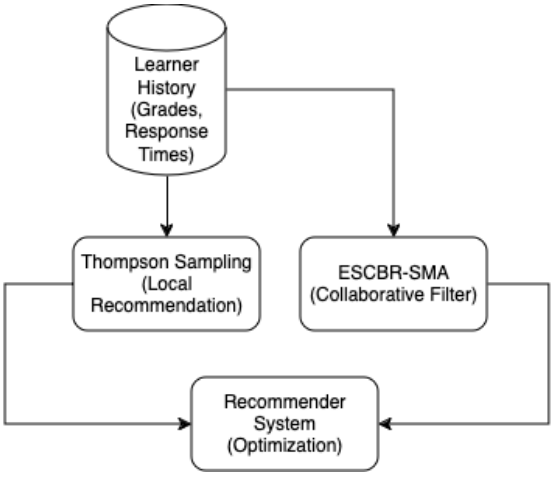

FIGURE 10.1 – Architecture Proposed Algorithm

La première étape est l'adaptation avec l'échantillonnage de Thompson et la prédiction de l'ECBR-SMA, et enfin la prise de décision pour l'envoi à l'apprenant. Le système de recommandation obtient une valeur de probabilité pour tous les niveaux de complexité pour l'apprenant et l'ECBR-SMA évalue la proposition avec une prédiction pour chaque niveau de complexité. Le tableau 10.1 présente les variables et les paramètres de l'algorithme proposé et les mesures employées.

L'intégration se fait en trois étapes. Tout d'abord, sont obtenus les valeurs aléatoires pour chaque niveau de complexité c en utilisant les distributions de probabilité générées avec le TS (équation 10.1), une fois que toutes les valeurs de probabilité correspondant à tous les niveaux de complexité ont été obtenues, la normalisation de toutes ces valeurs est calculée comme indiqué dans l'équation 10.2. Les valeurs de normalisation servent

ID	Type	Description	Domain
α	p	Beta distribution parameter	$[1, \infty] \in \mathbb{R}$
β	p	Beta distribution parameter	$[1, \infty] \in \mathbb{R}$
t	p	Time defined as iterations	$\mathbb{N}$
c	p	Complexity level	$\mathbb{N}$
x_c	p	Mean grades for complexity level c	$\mathbb{R}$
y_c	p	Number of questions for complexity level c	$\mathbb{N}$
$k_{t,c}$	v	Knowledge evolution in time t for complexity level c	$[0, 1] \in \mathbb{R}$
$vk_{t,c}$	v	Knowledge evolution for each complexity level c	$\mathbb{R}$
TS_c	v	Thompson sampling reward for a complexity level c	$[0, 1] \in \mathbb{R}$
TSN_c	v	Normalization of TS_c with others complexity levels	$[0, 1] \in \mathbb{R}$
$ESCBR_c$	v	Grade prediction for a complexity level c	$\mathbb{R}_+$
p_c	f	Probability density function for complexity level c	$\mathbb{R}_+$
D_{JS}	f	Jensen-Shannon divergence	$[0, 1] \in \mathbb{R}$
r	f	Recommender metric function	$[0, 1] \in \mathbb{R}$

TABLE 10.1 – Parameters (p), variables (v) and functions (f) of proposed algorithm and metrics

de paramètres de priorité pour les prédictions effectuées par l'ESCBR-SMA, comme le montre l'équation 10.3.

$$TS_c = rand(Beta(\alpha_c, \beta_c)) \tag{10.1}$$

$$TSN_c = \frac{TS_c}{\sum_{i=0}^{4} TS_i} \tag{10.2}$$

$$n_c = argmax_c(TSN_c * ESCBR_c) \tag{10.3}$$

Avec les valeurs finales calculées pour chaque niveau de complexité, le niveau de complexité qui a la valeur la plus élevée est proposé comme recommandation finale (équation 10.3).

Après la sélection du niveau de complexité, toutes les distributions de probabilité sont mises à jour selon le processus de Hawkes (équation 10.4) pour chaque paramètre α et β en utilisant la fonction d'intensité définie constante (équations 10.5 et 10.6) et la fonction d'excitation (équation 10.7). En simulant la courbe d'oubli dans lévolution de la distribution de probabilité Beta.

$$\lambda(t) = \mu(t) + \sum_{t_i<t} \phi(t - t_i) \tag{10.4}$$

$$\mu_{\alpha,c}(t) = \begin{cases} 2, & c = 0 \\ 1, & 1 \leq c \leq 4 \end{cases} \tag{10.5}$$

$$\mu_{\beta,c}(t) = \begin{cases} 1, & c = 0 \\ 3, & c = 1 \\ 5, & c = 2 \\ 7, & c = 3 \\ 9, & c = 4 \end{cases} \tag{10.6}$$

$$\phi_h(t) = (10)(0.02)e^{-0.02t} \tag{10.7}$$

Then, the equation 10.8 shows the complete definition for all α and the equation 10.9 the definition for β parameters.

$$\lambda_\alpha(t) = \mu_{\alpha,c}(t) + \sum_{t_i<t} \phi_h(t - t_i) \tag{10.8}$$

$$\lambda_\beta(t) = \mu_{\beta,c}(t) + \sum_{t_i<t} \phi_h(t - t_i) \tag{10.9}$$

And for each complexity level c, the equation 10.10 describe the distribution of probability.

$$P_c(x, \lambda_\alpha(t), \lambda_\beta(t)) = \frac{x^{\lambda_\alpha(t)}(1-x)^{\lambda_\beta(t)}}{\int_0^1 u^{\lambda_\alpha(t)}(1-u)^{\lambda_\beta(t)}du} \tag{10.10}$$

10.3/ RÉSULTATS ET DISCUSSION

10.3.1/ SYSTÈME DE RECOMMANDATION AVEC UNE BASE DE DONNÉES D'ÉTUDIANTS RÉELS (TS AVEC HAWKES)

Le système de recommandation TS a été testé avec un ensemble de données adapté extrait des données réelles des interactions des étudiants avec un environnement d'apprentissage virtuel pour différents cours [**?**], le total des apprenants est 23366, dans cette base de données il y a les notes de l'apprenant dans différents cours et de multiples types d'évaluations. Il s'agit du troisième test, pour lequel le format de la base de données est adapté à la structure AI-VT (notes, temps de réponse et niveau de complexité), les niveaux de complexité sont divisés en cinq étapes et calculés avec le pourcentage de poids défini dans l'ensemble de données. La figure 10.2 a été générée après 100 exécutions de l'algorithme et montre que même la stochasticité, l'algorithme est stable parce que la variance globale dans tous les niveaux de complexité est faible en fonction du nombre total d'apprenants et du nombre total de recommandations.

L'algorithme recommande plus de niveaux de faible complexité avec Hawkes, parce que la connaissance tend à diminuer avec le temps, puis l'algorithme force à renforcer la connaissance dans tous les niveaux de complexité et puisque la configuration initiale donne une plus grande probabilité aux niveaux inférieurs, l'algorithme tend à répéter les niveaux plus accessibles nécessaires pour atteindre les niveaux supérieurs.

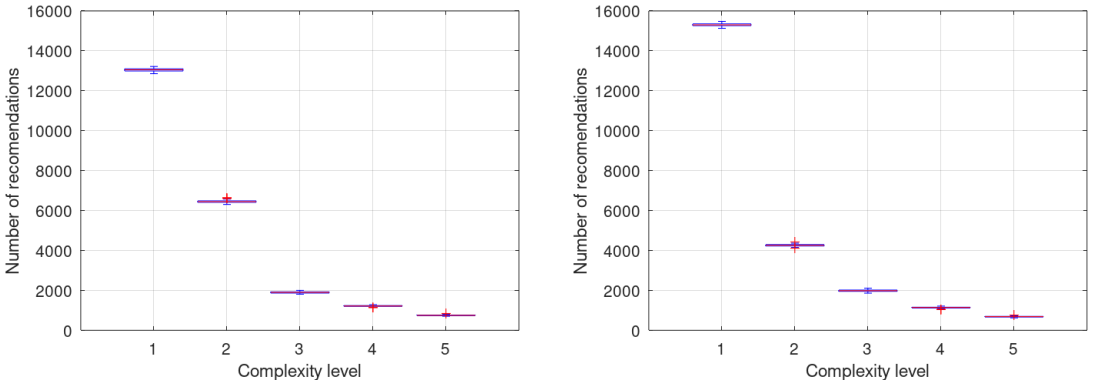

FIGURE 10.2 – Number of recommendations by complexity level (left static learning process, right dynamic learning process with Hawkes process)

10.3.2/ BASE DE DONNÉES SIMULÉE (ESCBR, TS AVEC HAWKES)

La base de données simulée est générée à l'aide d'une distribution log-normale de probabilité pour simuler les notes de 1000 apprenants dans cinq niveaux de complexité, chacun comportant quinze questions. Le générateur simule une plus grande complexité en réduisant la moyenne et en augmentant la variabilité de la distribution de probabilité.

La comparaison est faite avec la métrique décrite dans l'équation 10.11, où est calculée la relation entre la moyenne des notes x_c et le nombre de questions y_c pour chaque niveau de complexité c.

$$r(x_c, y_c) = e^{-2(x_c+y_c-1)^2} \tag{10.11}$$

Un scénario spécifique a été défini sans données initiales (notes et temps de réponse), c'est-à-dire un démarrage à froid. Le tableau 10.2 montre les résultats numériques après 10000 exécutions (1000 apprenants) pour TS et TS-Hawkes, dans le scénario évalué. Malgré les changements dans chaque niveau de complexité, le pourcentage total est très similaire.

	r_{C0}	r_{C1}	r_{C2}	r_{C3}	r_{C4}	Total	Percent
TS	0.9695	0.7945	0.6377	0.5615	0.5184	3.4816	69.632
TS-Hawkes	0.9414	0.7175	0.6434	0.5761	0.5448	3.4232	68.464

TABLE 10.2 – Metric comparison ESCBR-TS and ESCBR-TS-Hawkes

L'évolution de la variance (figure 10.3) montre qu'avec le processus de Hawkes, les valeurs sont maintenues autour de la configuration initiale, ce qui permet une plus grande adaptabilité aux changements dynamiques des connaissances qui se produisent dans le processus d'apprentissage. Étant donné que la distribution de probabilité Beta converge rapidement vers une valeur unique, plus on obtient de valeurs, plus la variance est faible et s'il y a un changement dans la valeur de convergence, la distribution a besoin de plus de données pour converger vers la nouvelle valeur, puisque les changements dans la moyenne sont proportionnels à la valeur de la variance avec un pas de changement constant pour les paramètres. Dans le cas de la modélisation du processus d'apprentissage, il est donc préférable de maintenir une valeur de variance relativement élevée pour

faciliter l'adaptation aux changements imprévus, ce qui constitue la principale contribution du processus de Hawkes à l'échantillonnage de Thompson pour la modélisation de l'évolution des connaissances.

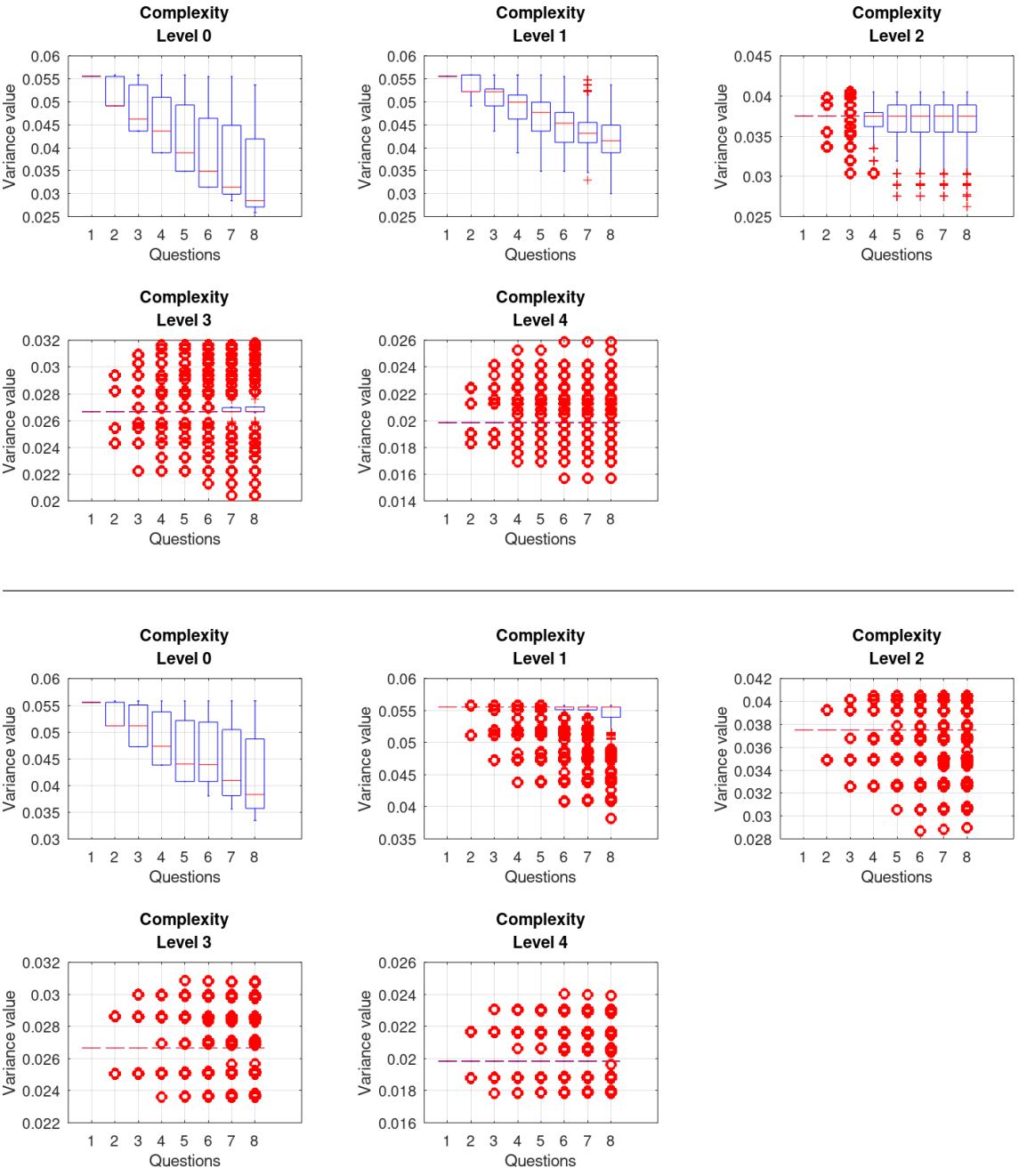

FIGURE 10.3 – Variance evolution for Beta distribution of probability and all complexity levels (Top : static learning process. Bottom : dynamic learning process with Hawkes process)

10.4/ CONCLUSION

Ce chapitre a présenté un algorithme intégré entre deux modèles développés précédemment, un système de recommandation basé sur l'algorithme d'échantillonnage de Thompson, un modèle de régression d'ensemble basé sur le raisonnement par cas et une simulation de courbe d'oubli en utilisant le processus de Hawkes. L'algorithme intégré est appliqué à un EIAH appelé AI-VT. Les résultats montrent en effet que l'intégration permet d'obtenir des résultats similaires, mais avec un processus plus réaliste, ce qui permet de mieux personnaliser le système et de faciliter l'acquisition de connaissances.

Les avantages du modèle proposé sont les suivants : i) il permet de générer des recommandations personnalisées pour chaque apprenant avec relativement peu de données historiques, ii) étant donné que plusieurs points de vue (différents algorithmes) sur le même problème et avec la même base de données sont intégrés sur la base du paradoxe de Stein, le risque de tomber dans les paradoxes de Simpson est réduit, iii) les deux modèles avec le processus de Hawkes sont plus réalistes et dynamiques dans le processus d'apprentissage global.

11

CONCLUSIONS ET PERSPECTIVES

11.1/ CONCLUSION GÉNÉRALE

Dans le travail réalisé pendant cette thèse.

11.2/ PERSPECTIVES

Le système devrait être testé avec des données réelles dans un environnement standard réel afin d'obtenir des informations sur certains groupes caractéristiques spécifiques d'étudiants et de comparer les résultats obtenus avec les résultats théoriques, consolidant ainsi les preuves comportementales et l'efficacité du modèle proposé.

Comme travail futur, il est proposé d'intégrer dans le modèle d'autres variables obtenues avec des algorithmes d'intelligence artificielle complémentaires tels que l'analyse vidéo, l'analyse audio et même l'analyse des données obtenues des apprenants tout au long du processus d'apprentissage. Analyser le modèle avec différentes configurations paramétriques, afin de déterminer quelles sont les configurations les plus appropriées et comment chaque variable influence le comportement global des algorithmes exécutés et le résultat final.

Comme perspectives de ce travail peuvent être envisagées :

- L'étude du modèle de raisonnement à partir de cas en profondeur avec l'analyse de changement des paramètres.
- Analyser le comportement de l'algorithme avec des metriques de distance différentes de la distance Euclidiene.
- Analyser la modification de la fonction de optimisation associé au choix de la meilleure solution dans les solutions générées par le second stacking processus.
- Détérminer le point de rupture pour différents scénarios.

Pour l'échantillonage de Thompson les perspectives ouvertes par ce travail sont :

- Changer la famille des distributions de probabilité et étudier la performace obtenue.
- Calculer de façon dynamique les taux corrélées d'actualisation des ditributions de probabilité pour chaque niveau de complexité.
- Ajouter une comportement décroissante à l'actualisation des distributions de probabilité pour améliorer l'adaptabilité su modèle.

Aussi, évaluer le système avec une base de données avec des erreurs pour mesurer la résilience du modèle et comment il peut s'autocorregir face à d'eventuels disfonctionnements.

De façon générale il est possible de définir des intervales pour chaque paramètre et executer un analyse de sensibilité pour évaluer les résultats avec l'objetif de estimer la stabilité du système et les limites numériques.

12

PUBLICATIONS

Pendant le travail de cette thèse, plusieurs articles ont été produits en explicitant les contributions réalisées, en parallèle d'autres publications ont été produites dans le cadre de la recherche en informatique.

12.1/ PUBLICATIONS PAR RAPPORT AU SUJET DE THÈSE

Daniel Soto Forero ; Julien Henriet ; Marie-Laure Betbeder. Ensemble Stacking Case-Based Reasoning, Stochastic Recommander Model and Hawkes process Applied to ITS AI-VT. 2025.

Daniel Soto Forero ; Julien Henriet ; Marie-Laure Betbeder. Ensemble Stacking Case-Based Reasoning and Stochastic Recommander Model Applied to ITS AI-VT. 2025.

Daniel Soto Forero ; Julien Henriet ; Marie-Laure Betbeder. Integration of Stacking Case Based Reasoning with Multi-Agents System Applied to Regression Problems. 2025.

Daniel Soto Forero ; Simha Ackermann ; Marie-Laure Betbeder ; Julien Henriet. The Intelligent Tutoring System AI-VT with Case-Based Reasoning and Real Time Recommender Models. International Conference on Case Based Reasoning (ICCBR). 2024.

Daniel Soto Forero ; Marie-Laure Betbeder ; Julien Henriet. Ensemble Stacking Case-Based Reasoning for Regression. International Conference on Case Based Reasoning (ICCBR). 2024.

Daniel Soto Forero ; Simha Ackermann ; Marie-Laure Betbeder ; Julien Henriet. Automatic Real-Time Adaptation of Training Session Difficulty Using Rules and Reinforcement Learning in the AI-VT ITS. International Journal of Modern Education and Computer Science (IJMECS). 2024.

12.2/ AUTRES PUBLICATIONS

Daniel Soto Forero ; Yony Ceballos. Metaheuristics Hybridation and Parametrization using Machine Learning. 2025.

Daniel Soto Forero ; Wilson Soto Forero. Efficient Exploration in Unknown Environments : An Adaptative Multi-Agent Approach. 19th Iberian Conference on Information Systems and Technologies. 2024.

Daniel Soto Forero ; Yony Ceballos. Stochastic agent-based models optimization applied to the problem of rebalancing bike-share systems. International Journal of Electrical and Computer Engineering. 2024.

Daniel Soto Forero ; Wilson Soto Forero. A Multi-Agent Based Algorithm for Quadratic Assignment Problem. IEEE Latin American Conference on Computational Intelligence. 2023.

BIBLIOGRAPHIE

[Aamodt and Plaza, 1994] Aamodt, A. and Plaza, E. (1994). Case-based reasoning : Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1) :39–59.

[Alabdulrahman and Viktor, 2021] Alabdulrahman, R. and Viktor, H. (2021). Catering for unique tastes : Targeting grey-sheep users recommender systems through one-class machine learning. *Expert Systems with Applications*, 166 :114061.

[Auer et al., 2021] Auer, F., Lenarduzzi, V., Felderer, M., and Taibi, D. (2021). From monolithic systems to microservices : An assessment framework. *Information and Software Technology*, 137 :106600.

[Butdee and Tichkiewitch, 2011] Butdee, S. and Tichkiewitch, S. (2011). Case-based reasoning for adaptive aluminum extrusion die design together with parameters by neural networks. In Bernard, A., editor, *Global Product Development*, pages 491–496, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Chiu et al., 2023] Chiu, T. K., Xia, Q., Zhou, X., Chai, C. S., and Cheng, M. (2023). Systematic literature review on opportunities, challenges, and future research recommendations of artificial intelligence in education. *Computers and Education : Artificial Intelligence*, 4 :100118.

[C.K. and R.C., 1989] C.K., R. and R.C., S. (1989). *Inside Case-Based Reasoning*. Psychology Press.

[Cunningham and Delany, 2021] Cunningham, P. and Delany, S. J. (2021). K-nearest neighbour classifiers - a tutorial. *ACM Comput. Surv.*, 54(6).

[Ezaldeen et al., 2022] Ezaldeen, H., Misra, R., Bisoy, S. K., Alatrash, R., and Priyadarshini, R. (2022). A hybrid e-learning recommendation integrating adaptive profiling and sentiment analysis. *Journal of Web Semantics*, 72 :100700.

[Feely et al., 2020] Feely, C., Caulfield, B., Lawlor, A., and Smyth, B. (2020). Using case-based reasoning to predict marathon performance and recommend tailored training plans. In Watson, I. and Weber, R., editors, *Case-Based Reasoning Research and Development*, pages 67–81, Cham. Springer International Publishing.

[Grace et al., 2016] Grace, K., Maher, M. L., Wilson, D. C., and Najjar, N. A. (2016). Combining cbr and deep learning to generate surprising recipe designs. In Goel, A., Díaz-Agudo, M. B., and Roth-Berghofer, T., editors, *Case-Based Reasoning Research and Development*, pages 154–169, Cham. Springer International Publishing.

[Gupta et al., 2021] Gupta, S., Chaudhari, S., Joshi, G., and Yağan, O. (2021). Multi-armed bandits with correlated arms. *IEEE Transactions on Information Theory*, 67(10) :6711–6732.

[Hajduk et al., 2019] Hajduk, M., Sukop, M., and Haun, M. (2019). *Cognitive Multi-agent Systems : Structures, Strategies and Applications to Mobile Robotics and Robosoccer*. Studies in Systems, Decision and Control. Springer International Publishing.

[Henriet et al., 2017] Henriet, J., Christophe, L., and Laurent, P. (2017). Artificial intelligence-virtual trainer : An educative system based on artificial intelligence and

designed to produce varied and consistent training lessons. *Proceedings of the Institution of Mechanical Engineers, Part P : Journal of Sports Engineering and Technology*, 231(2) :110–124.

[Henriet and Greffier, 2018] Henriet, J. and Greffier, F. (2018). Ai-vt : An example of cbr that generates a variety of solutions to the same problem. In Cox, M. T., Funk, P., and Begum, S., editors, *Case-Based Reasoning Research and Development*, pages 124–139, Cham. Springer International Publishing.

[Hoang, 2018] Hoang, L. (2018). *La formule du savoir. Une philosophie unifiée du savoir fondée sur le théorème de Bayes*. EDP Sciences.

[Hu et al., 2025] Hu, B., Ma, Y., Liu, Z., and Wang, H. (2025). A social importance and category enhanced cold-start user recommendation system. *Expert Systems with Applications*, 277 :127130.

[Huang et al., 2023] Huang, A. Y., Lu, O. H., and Yang, S. J. (2023). Effects of artificial intelligence–enabled personalized recommendations on learners' learning engagement, motivation, and outcomes in a flipped classroom. *Computers and Education*, 194 :104684.

[Ingkavara et al., 2022] Ingkavara, T., Panjaburee, P., Srisawasdi, N., and Sajjapanroj, S. (2022). The use of a personalized learning approach to implementing self-regulated online learning. *Computers and Education : Artificial Intelligence*, 3 :100086.

[Jean-Daubias, 2011] Jean-Daubias, S. (2011). Ingénierie des profils d'apprenants.

[Jung et al., 2009] Jung, S., Lim, T., and Kim, D. (2009). Integrating radial basis function networks with case-based reasoning for product design. *Expert Systems with Applications*, 36(3, Part 1) :5695–5701.

[Kolodner, 1983] Kolodner, J. L. (1983). Reconstructive memory : A computer model. *Cognitive Science*, 7(4) :281–328.

[Lalitha and Sreeja, 2020] Lalitha, T. B. and Sreeja, P. S. (2020). Personalised self-directed learning recommendation system. *Procedia Computer Science*, 171 :583–592. Third International Conference on Computing and Network Communications (CoCoNet'19).

[Leikola et al., 2018] Leikola, M., Sauer, C., Rintala, L., Aromaa, J., and Lundström, M. (2018). Assessing the similarity of cyanide-free gold leaching processes : A case-based reasoning application. *Minerals*, 8(10).

[Lepage et al., 2020] Lepage, Y., Lieber, J., Mornard, I., Nauer, E., Romary, J., and Sies, R. (2020). The french correction : When retrieval is harder to specify than adaptation. In Watson, I. and Weber, R., editors, *Case-Based Reasoning Research and Development*, pages 309–324, Cham. Springer International Publishing.

[Lin, 2022] Lin, B. (2022). Evolutionary multi-armed bandits with genetic thompson sampling. In *2022 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8.

[Maher and Grace, 2017] Maher, M. L. and Grace, K. (2017). Encouraging curiosity in case-based reasoning and recommender systems. In Aha, D. W. and Lieber, J., editors, *Case-Based Reasoning Research and Development*, pages 3–15, Cham. Springer International Publishing.

[Malburg et al., 2024] Malburg, L., Hotz, M., and Bergmann, R. (2024). Improving complex adaptations in process-oriented case-based reasoning by applying rule-based adaptation. In Recio-Garcia, J. A., Orozco-del Castillo, M. G., and Bridge, D., editors, *Case-Based Reasoning Research and Development*, pages 50–66, Cham. Springer Nature Switzerland.

[Muangprathub et al., 2020] Muangprathub, J., Boonjing, V., and Chamnongthai, K. (2020). Learning recommendation with formal concept analysis for intelligent tutoring system. *Heliyon*, 6(10) :e05227.

[Müller and Bergmann, 2015] Müller, G. and Bergmann, R. (2015). Cookingcake : A framework for the adaptation of cooking recipes represented as workflows. In *International Conference on Case-Based Reasoning*.

[Nkambou et al., 2010] Nkambou, R., Bourdeau, J., and Mizoguchi, R. (2010). *Advances in Intelligent Tutoring Systems*. Springer Berlin, Heidelberg, 1 edition.

[Obeid et al., 2022] Obeid, C., Lahoud, C., Khoury, H. E., and Champin, P. (2022). A novel hybrid recommender system approach for student academic advising named cohrs, supported by case-based reasoning and ontology. *Computer Science and Information Systems*, 19(2) :979–1005.

[Ontañón et al., 2015] Ontañón, S., Plaza, E., and Zhu, J. (2015). Argument-based case revision in cbr for story generation. In Hüllermeier, E. and Minor, M., editors, *Case-Based Reasoning Research and Development*, pages 290–305, Cham. Springer International Publishing.

[Parejas-Llanovarced et al., 2024] Parejas-Llanovarced, H., Caro-Martínez, M., del Castillo, M. G. O., and Recio-García, J. A. (2024). Case-based selection of explanation methods for neural network image classifiers. *Knowledge-Based Systems*, 288 :111469.

[Petrovic et al., 2016] Petrovic, S., Khussainova, G., and Jagannathan, R. (2016). Knowledge-light adaptation approaches in case-based reasoning for radiotherapy treatment planning. *Artificial Intelligence in Medicine*, 68 :17–28.

[Richter and Weber, 2013] Richter, M. and Weber, R. (2013). *Case-Based Reasoning (A Textbook)*. Springer-Verlag GmbH.

[Richter, 2009] Richter, M. M. (2009). The search for knowledge, contexts, and case-based reasoning. *Engineering Applications of Artificial Intelligence*, 22(1) :3–9.

[Robertson and Watson, 2014] Robertson, G. and Watson, I. D. (2014). A review of real-time strategy game ai. *AI Mag.*, 35 :75–104.

[Roldan Reyes et al., 2015] Roldan Reyes, E., Negny, S., Cortes Robles, G., and Le Lann, J. (2015). Improvement of online adaptation knowledge acquisition and reuse in case-based reasoning : Application to process engineering design. *Engineering Applications of Artificial Intelligence*, 41 :1–16.

[Sadeghi Moghadam et al., 2024] Sadeghi Moghadam, M. R., Jafarnejad, A., Heidary Dahooie, J., and Ghasemian Sahebi, I. (2024). A hidden markov model based extended case-based reasoning algorithm for relief materials demand forecasting. *Mathematics Interdisciplinary Research*, 9(1) :89–109.

[Schank and Abelson, 1977] Schank, R. C. and Abelson, R. P. (1977). *Scripts, Plans, Goals and Understanding : an Inquiry into Human Knowledge Structures*. L. Erlbaum, Hillsdale, NJ.

[Seznec et al., 2020] Seznec, J., Menard, P., Lazaric, A., and Valko, M. (2020). A single algorithm for both restless and rested rotting bandits. In Chiappa, S. and Calandra, R., editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3784–3794. PMLR.

[Sinaga and Yang, 2020] Sinaga, K. P. and Yang, M.-S. (2020). Unsupervised k-means clustering algorithm. *IEEE Access*, 8 :80716–80727.

[Skittou et al., 2024] Skittou, M., Merrouchi, M., and Gadi, T. (2024). A recommender system for educational planning. *Cybernetics and Information Technologies*, 24(2) :67–85.

[Smyth and Cunningham, 2018] Smyth, B. and Cunningham, P. (2018). An analysis of case representations for marathon race prediction and planning. In Cox, M. T., Funk, P., and Begum, S., editors, *Case-Based Reasoning Research and Development*, pages 369–384, Cham. Springer International Publishing.

[Smyth and Willemsen, 2020] Smyth, B. and Willemsen, M. C. (2020). Predicting the personal-best times of speed skaters using case-based reasoning. In Watson, I. and Weber, R., editors, *Case-Based Reasoning Research and Development*, pages 112–126, Cham. Springer International Publishing.

[Soto-Forero et al., 2024] Soto-Forero, D., Ackermann, S., Betbeder, M.-L., and Henriet, J. (2024). The intelligent tutoring system ai-vt with case-based reasoning and real time recommender models. In Recio-Garcia, J. A., Orozco-del Castillo, M. G., and Bridge, D., editors, *Case-Based Reasoning Research and Development*, pages 191–205, Cham. Springer Nature Switzerland.

[Su et al., 2022] Su, Y., Cheng, Z., Wu, J., Dong, Y., Huang, Z., Wu, L., Chen, E., Wang, S., and Xie, F. (2022). Graph-based cognitive diagnosis for intelligent tutoring systems. *Knowledge-Based Systems*, 253 :109547.

[Supic, 2018] Supic, H. (2018). Case-based reasoning model for personalized learning path recommendation in example-based learning activities. In *2018 IEEE 27th International Conference on Enabling Technologies : Infrastructure for Collaborative Enterprises (WETICE)*, pages 175–178.

[Uysal and Sonmez, 2023] Uysal, F. and Sonmez, R. (2023). Bootstrap aggregated case-based reasoning method for conceptual cost estimation. *Buildings*, 13(3).

[Wang et al., 2021] Wang, F., Liao, F., Li, Y., and Wang, H. (2021). A new prediction strategy for dynamic multi-objective optimization using gaussian mixture model. *Information Sciences*, 580 :331–351.

[Wolf et al., 2024] Wolf, T. N., Bongratz, F., Rickmann, A.-M., Pölsterl, S., and Wachinger, C. (2024). Keep the faith : Faithful explanations in convolutional neural networks for case-based reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 5921–5929.

[Xu et al., 2021] Xu, S., Cai, W., Xia, H., Liu, B., and Xu, J. (2021). Dynamic metric accelerated method for fuzzy clustering. *IEEE Access*, 9 :166838–166854.

[Yu and Li, 2023] Yu, L. and Li, M. (2023). A case-based reasoning driven ensemble learning paradigm for financial distress prediction with missing data. *Applied Soft Computing*, 137 :110163.

[Yu et al., 2024] Yu, L., Li, M., and Liu, X. (2024). A two-stage case-based reasoning driven classification paradigm for financial distress prediction with missing and imbalanced data. *Expert Systems with Applications*, 249 :123745.

[Zhang. and Aslan, 2021] Zhang., K. and Aslan, A. B. (2021). Ai technologies for education : Recent research and future directions. *Computers and Education : Artificial Intelligence*, 2 :100025.

[Zhao et al., 2023] Zhao, L.-T., Wang, D.-S., Liang, F.-Y., and Chen, J. (2023). A recommendation system for effective learning strategies : An integrated approach using context-dependent dea. *Expert Systems with Applications*, 211 :118535.

[Zhou and Wang, 2021] Zhou, L. and Wang, C. (2021). Research on recommendation of personalized exercises in english learning based on data mining. *Scientific Programming*, 2021 :5042286.

[Zuluaga et al., 2022] Zuluaga, C. A., Aristizábal, L. M., Rúa, S., Franco, D. A., Osorio, D. A., and Vásquez, R. E. (2022). Development of a modular software architecture for underwater vehicles using systems engineering. *Journal of Marine Science and Engineering*, 10(4).

Document réalisé avec LaTeX et :
le style LaTeX pour Thèse de Doctorat créé par S. Galland — http://www.multiagent.fr/ThesisStyle
la collection de paquets `tex-upmethodology` — http://www.arakhne.org/tex-upmethodology/

Résumé :

L'objectif de ce travail de thèse est la prise en compte en temps réel du travail d'un apprenant pour une meilleure personnalisation d'AI-VT (Artificial Intelligence Virtual Trainer).

Mots-clés : Mot-clé 1, Mot-clé 2

Abstract:

This is the abstract in English.

Keywords: Keyword 1, Keyword 2

SPIM

■ École doctorale SPIM 16 route de Gray F - 25030 Besançon cedex

■ tél. +33 (0)3 81 66 66 02 ■ **ed-spim@univ-fcomte.fr** ■ www.ed-spim.univ-fcomte.fr

UBFC