2025, vol. 5, n° 1, 70-94 pages, DOI: 10.21494/ISTE.OP.2025.1300 ISTE OpenScience

Proposition d'une architecture utilisant le *trace clustering* pour recommander un parcours d'apprentissage : définition des couches de fouille de processus et de recommandation

An Architecture Using *Trace Clustering* to Recommend a Learning Path: Definition of Process Mining and Recommender Layers

Wiem Hachicha[1,2], Leila Ghorbel[1], Ronan Champagnat[2], Corinne Amel Zayani[1], Mourad Rabah[2], Samuel Nowakowski[3]

[1]MIRACL - Université de Sfax, Tunis Road Km 10 PB. 242, 3021 Sfax, Tunisie, prenom.nom@fss.usf.tn
[2]L3i - Université de La Rochelle, Avenue Michel Crépeau, 17 042 La Rochelle, France, nom.prenom@univ-lr.fr
[3]LORIA - Université de Lorraine, Campus Scientifique, 615 rue du jardin-botanique, 54 506 Vandœuvre-lès-Nancy, France, samuel.nowakowski@loria.fr

RÉSUMÉ. Les systèmes d'informations pédagogiques permettent d'observer les traces d'apprentissage des apprenants et de mener des analyses sur leurs pratiques ou de prédire leur réussite. Dans ces travaux, nous étudions comment la fouille de processus, qui permet d'extraire des modèles de comportement des utilisateurs dans un système d'information, peut être utilisée dans un système de recommandation contextuel. Nous nous concentrons plus particulièrement sur le *trace clustering* qui vise à regrouper des traces possédant des dynamiques proches. Nos apports portent sur : la définition d'une architecture pour la recommandation qui utilise le *trace clustering* et la caractérisation des styles d'apprentissage des regroupements identifiés. Nous validons notre proposition sur les données collectées d'un cours d'introduction à la programmation d'IHM.

ABSTRACT. Educational information systems make it possible to observe learners' learning traces and to carry out analyses of their behaviour or to predict their success. In this work, we study how Process Mining can be used in contextual recommender systems. We are focusing in particular on Trace Clustering, which aims to group together traces with similar dynamics. Our contributions concern the definition of an architecture for recommendation that uses Trace Clustering and the characterization of the learning styles of the identified groups. We validate our proposal on data collected from an introductory course in UI programming.

MOTS-CLÉS. SI pédagogique, fouille de processus, recommandation, trace clustering.
KEYWORDS. Intelligent Tutor System, Process Mining, Recommendation, Trace Clustering.

Introduction

Les systèmes d'information pédagogiques se sont rapidement développés avec la mise en place d'environnements numériques de travail. La quantité de données produites et de traces laissées par les utilisateurs de ces systèmes offre l'opportunité de fournir des tableaux de bord d'apprentissage et des analyses sur les apprenants [Cordier et al., 2013].

La personnalisation des apprentissages est devenue un facteur essentiel pour la réussite des apprenants. Plusieurs projets se sont développés afin de traiter cette problématique. Ces projets s'attaquent au problème de l'offre de formation et de parcours d'apprentissage personnalisé, avec un accompagnement des étudiants dans leur projet de formation et d'insertion professionnelle. Toutefois, peu d'outils sont capables de fournir des indicateurs pertinents afin de recommander des parcours personnalisés ou des actions de remédiation pour les étudiants en difficulté.

La découverte des parcours d'apprentissage reste un défi à relever dans le domaine pédagogique. Toutefois, dans des domaines connexes, on trouve des travaux approchants. En particulier, dans le domaine de la fouille de processus, des recherches se sont focalisées sur l'extraction de connaissances sur le parcours de l'utilisateur à partir de l'analyse des traces d'exécution réelles des utilisateurs (enregistrement des chemins de navigation) [Leblay et al., 2018].

Le parcours d'apprentissage revient à sélectionner et ordonner les activités à réaliser pour acquérir des connaissances et compétences. Ce parcours d'apprentissage correspond à un scénario pédagogique suivi par un apprenant. Il peut être modélisé par un processus métier. Ce type de processus possède la particularité d'être faiblement structuré. C'est-à-dire que l'utilisateur dispose de degrés de liberté importants. L'objectif est alors de déterminer pour chaque utilisateur ou par groupe d'utilisateurs le comportement adopté lors de l'utilisation du système.

Notre objectif est d'étudier la possibilité de définir une méthodologie de recommandation basée sur les processus extraits des traces utilisateurs et d'implémenter l'architecture logicielle correspondante. Pour atteindre cet objectif, nous proposons de construire un modèle utilisant la fouille de processus à partir des observations recueillies lors des expériences des utilisateurs précédents avec le système d'information pédagogique. Ce modèle représente l'enchaînement des étapes et leurs impacts sur les états du processus global et sera par la suite utilisé pour recommander l'étape la plus appropriée pour guider l'utilisateur ou l'apprenant actuel. Un premier défi consiste à classer les différentes trajectoires que nous aurons identifiées au sein de systèmes d'activité sélectionnés dans un contexte d'apprentissage, afin de pouvoir ensuite déterminer, de manière automatisée, à quelle trajectoire type correspond le parcours et le développement d'un apprenant dans un environnement numérique dépourvu de processus métiers bien identifiés. Un second défi consiste à utiliser cette trajectoire pour faire une recommendation personnalisée à l'apprenant considéré.

Notre approche suggère une voie corrective, si nécessaire. Dans [Ho et al., 2016], les auteurs ont introduit une méthode pour amener les utilisateurs à prendre les bonnes décisions en fonction de certaines informations extraites d'un modèle de données qui décrit les activités possibles à réaliser pour atteindre l'objectif et leur impact. Ce modèle de données est une entrée de la méthode, dans le sens où il est construit *a priori* par des experts du domaine. Dans notre cas, notre approche calcule le modèle de données à partir des informations disponibles avant de recommander à l'étape suivante.

Dans cet article, nous étendons l'architecture logicielle définie dans les travaux de [Ghorbel et al., 2015] en enrichissant la couche fouille de processus par une étape de *trace clustering* (regroupement de traces). Les contributions de cet article concernent la définition et la validation de l'étape de trace clustering, déjà présentées lors de la conférence INFORSID [Hachicha et al., 2023], ainsi que la définition de la couche de recommandation en :

— caractérisant les modèles extraits des regroupements en calculant la similarité sémantique entre les noms des activités extraites de chaque modèle de scéanrio d'apprentissage et les descriptions des styles d'apprentissage en utilisant les transformers ;
— définissant une mesure de similarité entre le style d'apprentissage d'un nouvel apprenant et les styles d'apprentissages des modèles de processus des regroupements identifiés lors de l'étape de trace clustering.

Dans les deux sections suivantes, nous présentons d'abord le domaine de la fouille de processus. Nous décrivons le principe, les contraintes, les techniques de fouilles et les critères de qualité pour estimer la

pertinence du modèle extrait et nous introduisons le *trace clustering*. Puis, nous présentons le domaine de la recommandation en soulignant les apports de la recommandation sociale et en introduisant la notion de similarité sémantique.

Nous faisons ensuite un état de l'art de l'utilisation de la fouille de processus dans le domaine de l'éducation et positionnons nos contributions par rapport à l'existant. Puis, nous proposons notre architecture qui s'appuie d'une part sur le *trace clustering* pour classer les parcours d'apprentissage et, d'autre part, la similarité sémantique pour caractériser un regroupement de traces et formuler une recommandation. Nous validons cette architecture sur les données d'un cours universitaire de programmation des IHM.

1. Fouille de processus

La fouille de processus est utilisée dans de nombreux domaines, par exemple : pour la modélisation des comportements des utlisateurs qui recherchent des informations dans une bibliothèque numérique [Trabelsi et al., 2019a], [Trabelsi et al., 2019b], dans le domaine médical afin de cartographier les processus healthcare [Pika et al., 2019], dans les réseaux sociaux pour modéliser les parcours utilisateurs [Li & De Carvalho, 2019]... L'objectif de la fouille de processus est de découvrir, superviser et améliorer des processus métier existants en extrayant de la connaissance à partir des journaux d'événements facilement disponibles dans les systèmes d'information actuels.

Chaque événement dans un tel journal fait référence à une activité, une étape bien définie d'un processus, et est lié à un cas particulier, une instance de processus. Les techniques de fouille de processus utilisent des informations supplémentaires telles que la ressource, une personne ou un équipement, qui exécute ou lance l'activité, l'horodatage de l'événement ou des éléments de données enregistrés avec l'événement (par exemple, la quantité commandée).

Les techniques de fouille de processus sont apparues au cours des années 1990. [Cook & Wolf, 1998] et [Agrawal et al., 1998] ont proposé des algorithmes de découverte de modèles de processus afin de pouvoir analyser une organisation ou comparer des exécutions de processus métier à partir des traces observées dans le sysètme d'information. [Aalst, 2016] a popularisé la fouille de processus et développé de nombreux algorithmes.

La fouille de processus a été initialement développée dans le contexte de la modélisation des processus métiers. Les premiers travaux se sont concentrés sur la découverte d'un modèle de processus. Un algorithme de découverte analyse un journal d'événements et construit un modèle à partir des relations de précédences entre activités. Parmi les algorithmes de découverte nous pouvons citer Inductive Miner, Fuzzy Miner, Heuristic Miner et Alpha Miner. La sortie de ces algorithmes est un modèle tel que les réseaux de Petri, BPMN...

D'autres travaux ont porté sur la vérification de la conformité. Il s'agit ici de comparer un modèle existant avec celui découvert dans les journaux d'événement. L'objectif est de déterminer si la réalité enregistrée dans les journaux d'événements est conforme vis-à-vis du modèle de référence et réciproquement, i.e. le modèle est conforme vis-à-vis de journaux observés.

1.1. *Journaux d'événements*

Un journal d'événements contient un ensemble d'événements. Chaque événement correspond à la réalisation d'une activité. Les informations minimales afin d'extraire des connaissances en utilisant la fouille de processus sont [Aalst, 2016] :

— un identifiant permettant de rattacher l'activité à un processus (CaseID) ;
— un identifiant de l'activité (Activity), et,
— un horodatage (Timestamp).

Chaque variant d'un processus est décrit par une séquence d'activités (par exemple $\langle Activity1\ ,\ Activity2\ ,\ Activity1\rangle$). L'ensemble du journal d'événements, nommé L, peut s'écrire sous la forme $L = [\langle Activity1\ ,\ Activity2\ ,\ Activity1\rangle^2,\ \langle Activity1\ ,\ Activity2,\ Activity3\rangle^1, \langle Activity1\ ,\ Activity1\rangle^3]$ où la multiplicité des séquences donne le nombre de fois que ce variant est présent dans le journal.

1.2. *Algorithmes de découverte*

Les algorithmes de découverte visent à extraire des modèles de processus à partir des informations contenues dans les journaux d'événements. Ils se concentrent sur les aspects contrôle. Les techniques de découvertes sont variées, mais toutes sont bâties sur les relations entre les activités dans les journaux d'événements. Ces activités sont ordonnées en fonction de leur instant d'apparition.

Les algorithmes observent en particulier les relations de causalité entre deux activités. Ils se basent sur l'intuition que si une activité a_1 se trouve toujours immédiatement après une activité a_2, alors il y a certainement une relation de causalité entre elles. Quatre relations sont considérées :

1. *Succession directe*, $a_1 > a_2$ s'il existe un variant tel que a_1 est immédiatement suivie par a_2 (p. ex. $L = [\langle ...\ ,\ a_1\ ,\ a_2\ ,\ ...\rangle^2]$) ;
2. *Causalité*, $a_1 \rightarrow a_2$, si $a_1 > a_2$ et $a_2 \not> a_1$ (p. ex. $L = [\langle ...\ ,\ a_1\ ,\ a_2\ ,\ ...\rangle^2, \langle ...\ ,\ a_1\ ,\ ...\ ,\ a_2\ ,\ ...\rangle^3]$) ;
3. *Parallèle*, $a_1 \| a_2$, si $a_1 > a_2$ et $a_2 > a_1$ (p. ex. $L = [\langle ...\ ,\ a_1\ ,\ a_2\ ,\ ...\rangle^2, \langle ...\ ,\ a_2\ ,\ a_1\ ,\ ...\rangle^3]$) ;
4. *Choix*, $a_1 \# a_2$, si $a_1 \not> a_2$ et $a_2 \not> a_1$ (p. ex. $L = [\langle ...\ ,\ a_1\ ,\ ...\rangle^2, \langle ...\ ,\ a_2\ ,\ ...\rangle^3]$) .

L'algorithme α [Aalst et al., 2004] construit un raisonnement sur ces quatre relations pour déduire un réseau de Petri qui modélise les processus. D'autres algorithmes, tel qu'*Heuristic Miner* se basent sur le graphe des successions directes observées dans les journaux [Weijters & Ribeiro, 2011]. *Heuristic Miner* vise à résoudre le problème de logs contenant du bruit et obtenus à partir de processus faiblement structurés. Cet algorithme utilise une métrique basée sur la fréquence afin de déterminer la confiance dans la relation de causalité entre deux acitvités, a_1 et a_2 calculée comme suit :

$$a_1 \Rightarrow a_2 = \left(\frac{|a_1 > a_2| - |a_2 > a_1|}{|a_1 > a_2| + |a_2 > a_1| + 1} \right) \qquad [1.1]$$

Une autre stratégie de découverte, utilisée par *Fuzzy Miner* [Aalst, 2016], consiste à travailler sur la visualisation d'un graphe en s'inspirant de ce qui est fait pour les cartes routières et en donnant des règles de zoom et d'agrégations de chemins en fonction de leur popularité.

1.3. *Qualité des modèles découverts*

L'objectif de la fouille de processus est de découvrir des modèles qui caractérisent la dynamique d'un système d'information à partir des traces. Plusieurs mesures ont été proposées afin de caractériser la qualité du modèle découvert. Ces mesures font appel aux notions d'*overfitting*, le fait qu'un modèle ne représente que les variants présents dans les logs (un nouveau variant ne sera pas représenté par le modèle), et d'*underfitting*, le fait qu'un modèle représente potentiellement beaucoup (trop) de variants.

Quatre mesures sont définies par la communauté [Buijs et al., 2012] :
— *Fitness* : représente la capacité à expliquer les processus observés ;
— *Generalisation* : caractérise le fait que le modèle est capable de prendre en compte de nouveaux variants ;
— *Precision* : caractérise le fait que le modèle n'est pas trop général, c'est-à-dire qu'il ne prend pas en compte tous les variants ;
— *Simplicity* : caractérise la complexité et les spécificités du modèle.

Ces mesures visent à déterminer la capacité d'un modèle à expliquer et généraliser les dynamiques observées dans les journaux. Un bon modèle doit trouver un équilibre entre ces mesures. Pour chaque mesure plusieurs critères ont été proposés. Nous utiliserons ceux qui se basent sur la notion d'alignement entre une trace observée dans les logs et une trace d'exécution du modèle définis par [Adriansyah, 2014]. Nous nous intéressons particulièrement aux mesures de *Fitness*, de *Precision* et de *Generalisation*. En effet, la notion de simplicité est un mesure liée au modèle lui même et non pas à sa capacité à capturer un comportement observé.

Pour chaque trace, on totalise le nombre de transformations à faire pour passer de l'une à l'autre. La *Fitness* est calculée comme suit :

$$F(L, M) = 1 - \frac{\delta(\lambda^M_{opt}(L))}{\delta(\lambda^M_{worst}(L))} \quad [1.2]$$

Où δ est la fonction de coût, $\lambda^M_{worst}(L)$ est le cas le plus défavorable où il n'y a aucune synchronisation possible entre la trace et le modèle. $\lambda^M_{opt}(L)$ représente les coûts obtenus pour chaque alignement optimal.

La précision est calculée par :

$$P(T, M) = \frac{1}{|E|} \sum_{e \in E} \frac{en_T(e)}{en_M(e)} \quad [1.3]$$

Où E est l'ensemble des événements dans les logs T, A l'ensemble des activités, $en_T(e) \subseteq A$ est l'ensemble des activités présentes dans les traces et $en_M(e) \subseteq A$ l'ensemble des activités présentes dans le modèle.

La généralisation considère la fréquence avec laquelle chaque activité du processus est visitée. Plus les activités sont visitées, plus le modèle généralise le comportement modélisé. Nous calculons $P(e)$ comme la probabilité que la prochaine visite à l'état e révèle une activité non vue précédemment.

$$P(e) = \begin{cases} si\ N(e) \geq O(e) + 2 \Rightarrow \frac{O(e)(O(e)+1)}{N(e)(N(e)-1)} \\ sinon \Rightarrow 1 \end{cases} \quad [1.4]$$

Où la fonction $N(e)$ donne le nombre de fois où le système se trouve dans l'état e et $O(e)$ donne l'ensemble des évènements accessibles depuis l'état e

La généralisation est calculée par :

$$G(L, M) = 1 - \frac{\forall_{e \in \epsilon} P(e)}{|\epsilon|} \quad [1.5]$$

Où, ϵ désigne l'ensemble des événements rencontrés dans les logs.

1.4. *Trace clustering*

La fouille de processus est apparue dans le domaine des processus métiers. Ces processus sont généralement bien définis et cadrés au niveau du système d'information. Or, dans le cas de système d'information pédagogique, une partie du processus métier est entre les mains de l'apprenant. Nous avons donc des processus faiblement ou partiellement structurés. L'utilisation des algortimes de découverte dans un tel contexte amène à des modèles complexes difficilement exploitables.

Les travaux de [Trabelsi et al., 2021] montrent que l'utilisation du *trace clustering* aboutit à l'extraction de connaissance et permet d'identifier des dynamiques d'usages typiques en réduisant la complexité des modèles découverts. Le but du *trace clustering* est de regrouper des variants de scénarios d'apprentissage qui possèdent des caractéristiques, au niveau du processus décrit, similaires.

Le *trace clustering* consiste à regrouper les traces avant d'appliquer un algorithme de découverte [Diamantini et al., 2016]. Quatre approches ont été développées :

— le *trace-based clustering* regroupe les traces en fonction de leur similarité syntaxique. La mesure de similarité est inspirée de la distance de Levenshtein ;
— le *feature-based clustering* calcule un vecteur en fonction des caractéristiques de chaque trace. Parmi les caractérisitques couramment retenues, il y a la fréquence d'une activité dans un variant, la fréquence de succession directes, les sous-séquences maximales, les sous-séquences fréquentes...
— le *model-based clustering* qui réalise le regroupement sur les qualités des modèles minés, et
— l'*hybrid-based clustering* qui combine les approches précédentes [Song et al., 2008], [Zandkarimi et al., 2020].

Nous venons de présenter le domaine de la fouille de processus en terminant par une introduction au *trace clustering*. Il s'agit de méthodes visant à regrouper des traces d'exécutions qui décrivent des comportements similaires. Dans la section suivante, nous introduisons le domaine de la recommandation, avant de présenter un état de l'art des approches utilisant la fouille de processus dans un contexte d'apprentissage.

2. Systèmes de recommandation

Un système de recommandation trouve une partie de sa justification et de son utilité dans la prise en compte de la trajectoire propre de chaque apprenant au sein d'un système d'activités plus ou moins ouvert/contraint. Les systèmes de recommandation recueillent des informations afin de caractériser les profils des utilisateurs (préférences, intérêts...) pour un ensemble d'éléments tels qu'un film, un livre ou une ressource d'apprentissage. Ces informations peuvent être obtenues de manière explicite en enregistrant les évaluations des utilisateurs ou de manière implicite en observant le comportement des utilisateurs. Les systèmes de recommandation utilisent ces informations pour fournir aux utilisateurs des recommandations d'articles ou des prédictions [Bobadilla et al., 2013].

Les approches de recommandations sont classées en quatre grandes familles : basées sur le contenu, basées sur le filtrage collaboratif, hybrides et recommandation sociale [Al Fararni et al., 2020].

Les systèmes de recommandation basés sur le contenu proposent des articles similaires à ceux que l'utilisateur a préféré précédemment [Tang et al., 2013]. Ce type de système de recommandation analyse un ensemble d'articles évalués par des utilisateurs, puis construit un profil d'utilisateur basé sur ces descriptions. Il cherche alors la correspondance entre les attributs du profil de l'utilisateur avec les attributs d'un article pour formuler la recommandation.

La recommandation par filtrage collaboratif est basée sur le comportement de l'utilisateur ou sur l'évaluation par l'utilisateur des articles recommandés. Il recommande des articles appréciés par des utilisateurs au profil similaire et explore divers contenus possibles. À l'aide des informations contenues dans le profil de l'apprenant, le système de recommandation peut trouver des apprenants ayant des préférences d'apprentissage similaires et leur suggérer des ressources d'apprentissage en conséquence. L'algorithme de recommandation par filtrage collaboratif trouve soit des évaluations de prédiction, soit recommande une liste d'articles [Khanal et al., 2020].

Les approches hybrides visent à combiner les approches basées sur le contenu et celles de filtrage collaboratif pour surmonter leurs limites et bénéficier des forces des deux approches [Çano & Morisio, 2017].

Les approches précédentes font face à des limitations telles que le démarrage à froid ou la rareté des observations. Les systèmes de recommandation sociale sont apparus pour surmonter ces limitations. Dans un réseau social, les systèmes de recommandation sociale produisent des recommandations plus appropriées en fonction des relations sociales et des communautés [Shokeen & Rana, 2020]. En plus de la matrice d'évaluation utilisée par les autres systèmes de recommandation, les utilisateurs de ces systèmes sont connectés et fournissent des informations sociales [Tang et al., 2013]. Il existe trois types de systèmes de recomamndation sociale dans la littérature [Campana & Delmastro, 2017] : (i) les recommandations tenant compte des aspects sociaux [Abdelghani et al., 2018] lorsque les systèmes recommandent des personnes ou des amis, et incluent le concept de confiance et de relations sociales dans la recommandation d'articles génériques ; (ii) les recommandations basées sur les *tags* [Mezghani et al., 2017] lorsque les systèmes utilisent une certaine caractérisation des articles basée sur les *tags* pour recommander d'autres articles, *tags* et/ou personnes ; (iii) les recommandations tenant compte de la localisation [Kodama et al., 2009] lorsque les systèmes utilisent des informations liées à la localisation pour recommander des articles, des trajectoires et des personnes.

Les systèmes de recommandation se basent sur des mesures de similarité afin de déterminer quel article correspond le mieux à un utilisateur spécifique. Ces mesures fournissent une valeur qui correspond à la distance, ou similarité, entre deux utilisateurs ou articles. Les mesures de similarité couramment utilisées dans la littérature comprennent la distance euclidienne, la distance de Manhattan, la corrélation de Pearson, la similarité de Cosinus et la mesure de Jaccard.

Ces dernières années, les similarités sémantiques ont également été utilisées dans les systèmes de recommandation [Seidel et al., 2020]. Elles mesurent le degré d'équivalence sémantique entre deux éléments linguistiques, qu'il s'agisse de concepts, de phrases ou de documents [Zhang et al., 2015]. Ces mesures sont appliquées dans différents domaines tels que la recherche d'informations, le résumé de texte, l'analyse des sentiments, etc. Différentes méthodologies ont été proposées au fil des années pour mesurer la similarité sémantique, telles que les méthodes basées sur les réseaux neuronaux profonds comme les

transformers [Chandrasekaran & Mago, 2021]. Les architectures de transformeurs [Wolf et al., 2020], ont prouvé leurs performances pour la similarité textuelle sémantique, nous pouvons citer Bidirectional Encoder Representations from Transformers (BERT) [Devlin et al., 2018], Robustly optimized BERT approach (RoBERTa) [Liu et al., 2019], et une version distillée de BERT (DistilBERT) [Sanh et al., 2019].

Un transformeur est composé de deux parties principales l'encodeur et le décodeur. L'architecture complète comporte un nombre N d'encodeurs et un nombre N de décodeurs empilés.

Chaque encodeur réalise l'auto-attention et le traitement *feed-forward*. L'encodeur contient deux sous-couches : un *Multi-Head Attention* et un réseau *Feed Forward*. Il y a des connexions résiduelles autour de chacune des deux sous-couches suivies d'une normalisation de couche. La première sous-couche implémente un mécanisme d'auto-attention à plusieurs têtes (*Multi-Head Attention*) permet au modèle de se concentrer sur différentes parties du texte simultanément, capturant ainsi divers types de relations. La deuxième sous-couche est un réseau *Feed Forward* qui traite les informations à chaque position de manière indépendante, permettant une parallélisation efficace. Une normalisation de couche est effectuée après chaque sous-couche pour stabiliser le réseau, ce qui réduit le temps d'apprentissage nécessaire. Les *embeddings* sont utilisés pour convertir les jetons d'entrée et les jetons de sortie en vecteurs de dimension d. Le transformer n'a pas de notion d'ordre séquentiel. Un *Positional Encoding* consiste à ajouter de l'information sur la position relative ou absolue des jetons dans la séquence.

Chaque décodeur réalise l'auto-attention, l'attention multi-têtes sur la sortie de l'encodeur, et le traitement *feed-forward*. Le décodeur contient un mécanisme d'auto-attention qui est masqué pour empêcher le décodeur de s'occuper des mots suivants. Au niveau de la sous-couche *Multi-Head Attention*, le décodeur reçoit également la sortie de l'encodeur, ce qui permet au décodeur de s'occuper de tous les mots de la séquence d'entrée. Les sous-couches de décodeur *Multi-Head Attention* et réseau *Feed Forward* sont similaires à ceux implémentés dans les sous couches de l'encodeur.

3. Positionnement

Il existe de nombreuses techniques d'analyse de données issues de systèmes d'information pédagogiques telles que l'*Educational Data Mining* (EDM) [Berland et al., 2014] et l'*Educational Process Mining* (EPM) [Romero & Ventura, 2013]. L'*Educational Process Mining* permet de découvrir des modèles de processus d'apprentissage à partir de journaux d'événements à des fins différentes, telles que la prédiction des performances des apprenants, l'adaptation des contenus et la recommandation des ressources ou des parcours.

Les systèmes de recommandation doivent être adaptés aux domaines spécifiques qu'ils abordent. Dans le cas de l'éducation, les systèmes de recommandation jouent un rôle important pour les apprenants. L'objectif d'un système de recommandation pédagogique est de recommander à l'apprenant un contenu, un parcours ou une ressource d'apprentissage [Kolekar et al., 2019, Yan et al., 2021]. Les données d'entrée du système de recommandation pédagogique sont liées au profil de l'apprenant et contiennent diverses caractéristiques. Parmi les caractéristiques de l'apprenant utilisées pour la recommandation, nous citons les intérêts de l'apprenant [Dwivedi et al., 2018], les fichiers logs de l'apprenant [Kolekar et al., 2019], le style d'apprentissage [Nafea et al., 2019], etc.

La fouille de processus a été utilisée pour analyser les parcours d'apprentissage des étudiants.

[Beemt et al., 2018] explorent la relation entre le comportement d'apprentissage et les progrès d'apprentissage dans les MOOC afin de mieux comprendre comment les étudiants qui réussissent et ceux qui échouent répartissent leurs activités au cours des semaines de cours. Pour trouver les modèles de comportement d'apprentissage des étudiants dans le MOOC, une analyse des trajectoires d'apprentissage est réalisée en utilisant la fouille de processus. Le clustering hiérarchique est utilisé afin d'extraire la connaissance. Les auteurs ont trouvé quatre groupes d'étudiants significatifs, chacun représentant un comportement spécifique. Les techniques d'exploration de processus montrent que les étudiants qui réussissent affichent un comportement d'apprentissage plus régulier. Dans cette étude, le *trace clustering* est utilisé pour classer les scénarios d'apprentissage des étudiants.

Dans [Romero et al., 2008], une tâche de prétraitement est effectuée pour regrouper les utilisateurs en fonction de leur type d'interactions avec le cours. Cette étude permet de découvrir les comportements de navigation les plus spécifiques en utilisant uniquement les données groupées plutôt que le jeu de données complet en appliquant l'algorithme *Heuristic Miner*.

[Real et al., 2020] ont présenté les résultats de l'utilisation de techniques de fouille de processus pour valider les parcours d'apprentissage des étudiants dans un cours d'introduction à la programmation. Ils ont utilisé un journal d'événements Moodle contenant 24605 événements soumis par 73 étudiants de premier cycle. Les résultats ont révélé que, dans l'ensemble, les étudiants qui ont réussi et ceux qui ont échoué ont emprunté des chemins différents pour réaliser les activités du cours. Ils ont également obtenu les flux de contrôle et les fréquences des activités et des connexions pour identifier les dépendances et les ressources qui ont démarré ou terminé le processus. L'analyse de ces résultats fournit des informations générales et spécifiques sur les parcours d'apprentissage des étudiants, ainsi que la possibilité pour les enseignants d'observer les comportements et les progrès des étudiants.

[Martinez et al., 2021] ont examiné les trajectoires d'apprentissage des étudiants dans leurs études d'informatique. L'étude s'est concentrée sur la modélisation des caractéristiques qui influencent les taux d'abandon. Par conséquent, les trajectoires des étudiants ayant abandonné leurs études et de ceux qui les ont terminées sont analysées et comparées à l'aide d'outils de fouille de processus. Les auteurs ont constaté que les cours jugés difficiles et entravant la progression académique ont été identifiés, tout comme les derniers cours suivis par les étudiants avant l'abandon ou l'obtention du diplôme. Ils ont également constaté que les étudiants en décrochage terminent généralement après la première année et suivent des cours de programmation.

Ces travaux mettent en avant l'intérêt de la fouille de processus pour expliquer les parcours d'apprentissage. Toutefois ils ne permettent pas de classer les étudiants en fonction de leurs parcours.

Dans [Real et al., 2021], les auteurs ont présenté les résultats de l'utilisation de la fouille de processus et du Sequential Pattern Mining (SPM) pour vérifier et analyser les parcours d'apprentissage des étudiants dans un cours d'introduction à la programmation. Ils ont utilisé pour cela un journal d'événements de Moodle. Leur recherche a visé à déterminer quels types d'activité, quelles activités et quelles actions ont été réalisées et dans quel ordre. Les résultats ont révélé que les élèves ont eu des comportements distincts lorsqu'ils ont accédé aux activités et les ont exécutées. Les auteurs soulignent deux résultats qui ont permis de mieux comprendre les différents comportements : (i) la vérification des modèles de processus des attributs de niveaux 2 et 3 permet d'approfondir l'analyse du flux des événements des activités ; (ii) les sous-séquences d'intérêt extraites via SPM peuvent compléter l'analyse des comportements des

étudiants. Cette approche est intéressante car elle permet d'identifier des comportements spécifiques au cours du processus d'apprentissage. Toutefois, il n'est pas possible de classer les étudiants en fonction de leurs parcours.

La fouille de processus a aussi été utilisée avec un objectif de recommandation d'activités pédagogiques.

En utilisant la fonctionnalité de journalisation expérimentale de l'outil de modélisation *JMermaid* et les techniques de fouille de processus, [Sedrakyan et al., 2014] analysent le modèle de comportement (données d'événements de modélisation conceptuelle de 20 cas et 10000 événements). Les résultats de ce travail comprennent des modèles qui indiquent une performance d'apprentissage pire/meilleure. Les auteurs soutiennent que les résultats aident à améliorer les conseils d'enseignement pour la modélisation conceptuelle visant à fournir un feedback orienté vers le processus et fournissent des recommandations sur le type de données qui peuvent être utiles pour observer le comportement de modélisation du point de vue des résultats d'apprentissage.

Dans [Leblay et al., 2018], les auteurs ont cherché à proposer une méthode d'assistance basée sur le parcours d'apprentissage pour aider les apprenants à construire leur parcours universitaire. Ils utilisent la découverte de processus pour extraire les parcours d'apprentissage des apprenants précédents, puis utilisent ce modèle pour recommander l'étape la plus appropriée pour guider l'apprenant actuel.

La fouille de processus a été utilisée pour expliquer des comportements, analyser des parcours d'apprentissage et également avec un objectif de recommandation. Les derniers travaux sur la recommandation se basent sur l'analyse du modèle global d'apprentissage. La difficulté d'utiliser la découverte de modèle sur l'ensemble des traces collectées est que le modèle peut être complexe et il devient donc difficile d'interpréter les parcours.

Le *trace clustering* vise à regrouper des parcours d'utilisateurs similaires. Cela nous semble pertinent pour classer les scénarios d'apprentissage pour mieux orienter et cibler la recommandation.

Nous avons réalisé une étude comparative des travaux sur les systèmes de recommandation éducatifs, comme le montre le tableau 1. La comparaison repose sur une approche avec un système de recommandation, l'algorithme appliqué dans le système de recommandation, les données d'entrée, les données de sortie et la mesure de similarité. [Nafea et al., 2019] et [Yan et al., 2021] utilisent comme données d'entrée dans le système de recommandation le style de l'apprenant qui est insuffisant. [Kolekar et al., 2019] utilise comme données d'entrée le style de l'apprenant et les fichiers journaux de l'apprenant. Ces travaux présentent certaines lacunes en termes de données d'entrée. C'est pourquoi nous nous intéressons au style d'apprentissage et aux modèles de processus comme données d'entrée dans notre système de recommandation pédagogique. Les modèles de processus permettent de visualiser et de déduire le modèle de comportement réel de l'apprenant. Un style d'apprentissage est la façon dont un apprenant individuel préfère apprendre [Truong, 2016]. Il existe plusieurs modèles de style d'apprentissage [Felder & Silverman, 1988], [Kolb et al., 2007], [Myers, 1985].

Nos travaux de recherche proposent une architecture qui combine le regroupement des traces avec la puissance des transformeurs pour caractériser les modèles de processus découverts à partir des journaux d'événements éducatifs. Nous détaillons cette architecture dans la section qui suit.

Référence	Approche de recommandation	Algorithme utilisé	Données d'entrée	Données de sortie	Mesure de similarité
[Kolekar et al., 2019]	content-based	clustering	learner's style learner's log files	learning path learning content	no similarity measure
[Nafea et al., 2019]	hybrid	k-means	learner's style	rating	Pearson correlation cosine similarity
[Yan et al., 2021]	collaborative filtering	association rules mining	learner's style	learning resource	Euclidean distance
Our work	collaborative filtering	trace clustering	Learner's style process models	process models ranked	transformer

TABLEAU 1. *Comparaison des approches de recommandation pour l'éducation*

4. Architecture pour la recommandation

Notre architecture se base sur les travaux de [Ghorbel et al., 2015] qui traitent de l'adaptation de contenu pédagogiques pour la personnalisation des enseignements. Elle permet :

— l'échange de données entre des profils d'apprenants hétérogènes sur la base d'une couche d'interopérabilité, et,

— d'adapter aux apprenants la navigation dans les ressources d'apprentissage sur la base d'une couche d'adaptation.

Notre architecture, illustrée par la figure 1, est organisée en quatre couches :

— la couche *source* stocke les objets d'apprentissage et les données de profil ;

— la couche *fouille de processus* qui a pour but d'extraire les modèles de processus d'apprentissage ;

— la couche *recommandation* propose une adaptation du scénario pédagogique, et

— la couche *client* chargée des interactions utilisateur.

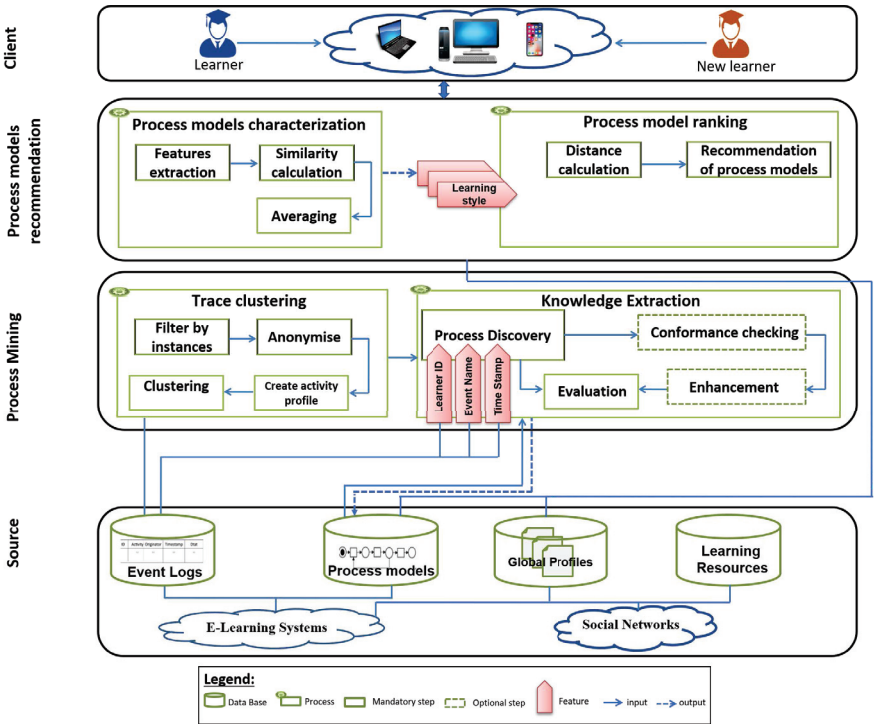

FIGURE 1. *Architecture logicielle pour la recommandation d'activités basée sur le trace clustering*

Nous étendons l'architecture présentée dans [Hachicha et al., 2021] avec deux étapes supplémentaires : une étape de *trace clustering* au niveau de la couche fouille de processus et une étape de caractérisation et d'évaluation des clusters au niveau de la couche recommandation.

Dans ce qui suit, nous introduisons brievement les couches client et source et présentons plus en détail les couches fouille de processus et recommandation.

4.1. *La couche client*

La couche client permet l'interaction entre l'apprenant et les systèmes d'apprentissage en ligne (*Learning Management Systems*, LMS). Elle représente l'interface utilisateur permettant d'intéragir avec le système. Ainsi, l'apprenant peut envoyer ses requêtes en cliquant sur les liens fournis à travers différents types de dispositifs (PC, mobile...).

4.2. *La couche source*

La couche source contient des bases de données distribuées de journaux d'événements, de modèles de processus, de profils globaux et de ressources d'apprentissage. Les journaux d'événements (*Event Logs*) sont des fichiers qui contiennent une grande quantité de données brutes sur l'interaction des apprenants avec le LMS. Comme les données dans les journaux d'événements sont souvent bruités, cette base de données nécessite une étape de prétraitement. Cette couche contient toutes les traces, laissées par l'utilisateur, que nous allons exploiter afin d'extraire son scénario d'apprentissage.

La base de données de *Global Profil* enregistre un profil pour chaque apprenant qui contient une vue globale de ses données qui sont distribuées dans les différents systèmes d'apprentissage en ligne. La construction du profil global est une étape préliminaire de la couche d'interopérabilité détaillée dans [Troudi et al., 2020]. Le profil contient plusieurs caractéristiques : des données personnelles, des données démographiques et des données sociales qui caractérisent les personnes connues (amis) et les intérêts (les intérêts de l'apprenant et ceux de ses amis). Les intérêts représentent les objets d'apprentissage sur lesquels l'apprenant prend plaisir à passer du temps à apprendre. Nous étendons ce profil avec des informations sur le type de situation d'apprentissage (à distance, en face à face ou hybride), l'aspect social (travail individuel ou collaboratif) et le style d'apprentissage (aspect théorique ou pratique).

La base de données des ressources d'apprentissage, *Learning Resources*, contient tout élément impliqué dans le processus d'apprentissage tel que les cours, quiz, pages web, images, vidéos...

4.3. *La couche fouille de processus*

La couche de fouille de processus permet de découvrir des modèles de processus d'apprentissage basés sur les journaux d'événements. Elle comprend une étape de *trace clustering* afin d'identifier des groupes homogènes de modèles d'apprenants (instances de processus possédant des caractéristiques proches). Nous nous sommes basés sur l'approche *feature-based clustering* en considérant la fréquence des activités par variant.

Notre contribution porte sur l'ajout de cette étape de *trace clustering*. Nous cherchons à regrouper les étudiants en fonction de leurs scénarios d'apprentissage préférentiels avant d'extraire des informations qui nous aiderons pour la couche de recommandation.

Comme le montre la figure 1, cette étape commence d'abord par le filtrage du journal des événements par instances pour nous assurer que seules les activités des apprenants soient conservées. Nous éliminons ainsi le bruit (activités d'administration du cours Moodle), les *outliers*, et les traces incomplètes (abandons en cours de formation).

Nous avons ensuite créé le profil d'activités, utilisé pour diviser le journal des événements. Le profil d'activités est une matrice $N \times M$, où N représente le nombre d'apprenants et M le nombre d'activités. Chaque ligne de cette matrice correspond à une trace vectorielle composée de fréquences d'activités.

Enfin, nous appliquons un algorithme de *clustering* pour diviser les traces en fonction du profil d'activités. Afin de sélectionner l'algorithme de *clustering* le plus adapté à notre cas nous avons réalisé des essais avec *DBSCAN*, *Agglomerative Clustering*, *Gaussian Mixture Model* et *k-means* (cf. section 5).

Les résultats du *trace clustering* constituent l'entrée de l'algorithme de découverte des processus. Ils sont constitués des traces d'un même regroupement qui correspondent aux scénarios pédagogiques suivis par les étudiants de ce regroupement. Il est alors possible d'appliquer les algorithmes de découvertes de processus afin d'obtenir le modèle de scénario pédagogique pour un groupe.

Chaque modèle de processus montre le comportement d'utilisation le plus courant des apprenants dans un LMS. L'analyse du modèle de processus permet de visualiser et de reproduire le comportement réel de l'apprenant, de trouver des *patterns* dans le comportement d'apprentissage des apprenants, et plus encore de proposer une explication sur le scénario d'apprentissage et ainsi de la recommandation faite.

4.4. *La couche recommandation*

Nous venons de voir que la couche de la fouille de processus permet la découverte de modèles de processus basée sur le journal d'événements. Les modèles de processus découverts constituent l'entrée de la couche de recommandation de modèles de processus.

La recommandation se fait en cours d'exécution et consiste à déterminer, à partir du modèle des processus et de la séquence réalisée jusque là par l'utilisateur, quelle activité doit être réalisée pour finir le processus en optimisant des critères préalablement définis.

La couche de recommandation vise à recommander à un nouvel apprenant un scénario pédagogique basé sur son style d'apprentissage et sur les modèles de processus des apprenants précédents. Elle se compose en deux étapes : l'extraction des caractéristiques des modèles de processus et le leur classement.

4.4.1. Extraction des caractéristiques des modèles de processus

Cette étape vise à caractériser chaque modèle de processus en fonction du style d'apprentissage. Cette caractérisation repose sur la sémantique ainsi que la fréquence des activités présentes dans chaque modèle de processus découvert. Cela permet de déterminer la proximité entre chaque style d'apprentissage et tous les noms d'activités. Ce module se compose de trois phases.

1. « Extraction de caractéristiques » consiste à extraire de chaque modèle de processus les noms des activités et leurs fréquences.
2. « Calcul de similarité », vise à calculer la somme des valeurs de similarité sémantique entre chaque style d'apprentissage dls_k et chaque nom d'activité $nameAct_i$ extrait du modèle de processus PM_j, en prenant en compte la fréquence de cette activité $freqAct_i$. Nous proposons d'utiliser les transformeurs, qui ont démontré au cours des dernières années leur capacité à identifier la sémantique des données. Par conséquent, il est plus évident d'identifier les styles d'apprentissage d'un modèle de processus à partir de noms d'activités.
3. Enfin, « Calcul de moyenne », permet de calculer la moyenne des valeurs de similarité sémantique obtenues à partir de la phase précédente de « calcul de similarité » (cf. équation 4.6).

$$V(PM_j, dls_k) = \frac{\sum_{i=1}^{n} sim_i\,(nameAct_i, dls_k) * freqAct_i}{\sum_{i=1}^{n} freqAct_i} \quad [4.6]$$

4.4.2. Classement des modèles de processus

Le but de cette étape est de recommander des modèles de processus à l'apprenant du plus pertinent au moins pertinent. Elle est composée de deux phases : « Calcul des distances » et « Recommandation de modèles de processus ».

La phase de « calcul des distances » a pour but de calculer la distance entre l'apprenant, caractérisé par son style d'apprentissage, et chaque modèle de processus PM_j en fonction du style d'apprentissage $V(PM_j, ls_k)$ obtenu lors de l'étape de « caractérisation des modèles de processus » (cf. 4.4.1).

L'algorithme 1 est exécuté lors de la phase « Recommandation de modèles de processus ». Il prend en entrées les descriptions des styles d'apprentissage *learningStyleDesc* et la liste des modèles de processus *listPM*. Le résultat généré par cet algorithme correspond à une liste ordonnée des modèles de processus nommée *listRecPM*.

Cet algorithme exécute, en premier lieu, la fonction intitulée *characterizeProcessModels* qui fait référence à l'étape « caractérisation des modèles de processus ». Cette fonction fait appel à deux autres fonctions : *extractFromPM* et *averageCalculation*. La fonction *extractFromPM* (ligne 6) est relative à la phase « extraction de caractéristiques ». Elle permet d'extraire à partir de chaque modèle de processus *p* de la liste *listPM* les noms des activités *namesAct* et leurs fréquences *freqAct*.

La fonction *averageCalculation* (lignes 17-25), correspond à la phase « calcul de moyenne ». Cette fonction prend en entrée les descriptions des styles d'apprentissage *learningStyleDesc*, les noms des activités *namesAct* et leurs fréquences *freqAct*. Elle calcule la somme des valeurs de similarité sémantique entre chaque style d'apprentissage et les noms des activités en appelant la fonction *semanticSimilarityCalculation* et en tenant compte des fréquences des activités. Cette fonction génère en sortie les valeurs moyennes de chaque style d'apprentissage, appelées *avgLs*.

La fonction *semanticSimilarityCalculation* correspond à la phase « calcul de similarité » de l'étape « caractérisation des modèles de processus », qui repose sur un transformeur. Cette fonction (lignes 11-16) prend en entrée les descriptions des styles d'apprentissage *learningStyleDesc* et les noms des activités *namesAct*. Nous encodons la liste des noms des activités ainsi que la liste des descriptions des styles d'apprentissage pour obtenir leurs *embeddings* respectifs, *embAct* et *embLsDesc*. Puis, nous calculons les scores de similarité entre les deux *embeddings* pour obtenir la valeur de similarité notée *sim*.

Après avoir caractérisé les modèles de processus, l'algorithme exécute en second lieu la fonction *processModelsRanking* (lignes 26-32). Cette fonction correspond à l'étape « classement des modèles de processus ». Elle prend en entrée les valeurs du style d'apprentissage de l'apprenant *learnerLs* et la moyenne des styles d'apprentissage des différents modèles de processus *avgLs*. Cette fonction génère en sortie les modèles de processus recommandés *listRecPM*.

Elle fait appel à la fonction *calculateDistance*, qui prend en entrée le style d'apprentissage de l'apprenant *learnerLs* et celui de chaque modèle de processus *lsPM* de la liste *listAvgLs*. Sur la base des distances calculées *listDistance*, une liste ordonnée *listRecPM* de modèles de processus est générée grâce à l'appel de la fonction *recommendProcessModels*. Ces modèles de processus, qui seront recommandés à l'apprenant, sont classés du plus pertinent au moins pertinent.

Algorithm 1: Recommandation de modèles de processus

```
   Input  : learningStyleDesc : list ; listPM : ProcessModel ;
   Output: listRecPM : ProcessModel ;
1  Begin
2      listAvgLs ← characterizeProcessModels(learningStyleDesc, listPM);
3      listRecPM ← processModelsRanking(learnerLs, listAvgLs);
       // Caractériser les modèles de processus en fonction des styles d'apprentissage
4      Function characterizeProcessModels(learningStyleDesc, listPM)
5          foreach p ∈ listPM do
6              namesAct, freqAct ← extractFromPM(p);
7              listAvgLs ← averageCalculation(learningStyleDesc, namesAct, freqAct);
8          end
9          return listAvgLs;
10     End
       // Calculer la similarité sémantique entre les noms d'activités et les descriptions de styles
          d'apprentissage
11     Function semanticSimilarityCalculation(learningStyleDesc, namesAct)
12         embAct ← calculateEmbedding(namesAct);
13         embLsDesc ← calculateEmbedding(learningStyleDesc);
14         sim ← calculateSimilarityEmbeddings(embAct, embLsDesc);
15         return sim;
16     End
       // Calculer la moyenne des valeurs de similarité sémantique obtenues à partir de la fonction de
          semanticSimilarityCalculation
17     Function averageCalculation(learningStyleDesc, namesAct, freqAct)
18         foreach dls ∈ learningStyleDesc do
19             foreach n ∈ namesAct do
20                 total+ = semanticSimilarityCalculation(dls, n) ∗ freqAct(n);
21             end
22             avgLs ← total/freqAct;
23         end
24         return avgLs;
25     End
       // Classer les modèles de processus en fonction du style d'apprentissage de l'apprenant
26     Function processModelsRanking(learnerLs, ListAvgLs)
27         foreach lsPM ∈ listAvgLs do
               // Calculer la similarité entre le style d'apprentissage de l'apprenant et les différents
                  modèles de processus
28             listDistance ← calculateDistance(learnerLs, lsPM);
29         end
           // Recommander à l'apprenant des modèles de processus
30         listRecPM ← recommendProcessModels(listDistance);
31         return listRecPM;
32     End
33 End
```

5. Validation de l'architecture proposée

Afin de valider notre architecture nous avons extrait les journaux d'événements des apprenants ayant suivi sur un semestre le cours « Introduction aux interfaces homme-machine (IHM) » créé sur la plateforme Moodle à La Rochelle Université. Un scénario pédagogique a été établi permettant aux apprenants

d'atteindre l'objectif final. Les journaux d'événements ont été enregistrés sur deux années (2019-2020 et 2022-2023), donnant deux jeux de données (IHM_1 et IHM_2) avec, respectivement, 42 438 événements pour 100 étudiants (c'est-à-dire 100 traces) et 25 222 événements pour 50 étudiants. Chaque événement correspond à une activité réalisée par un apprenant.

Nous proposons le scénario suivant (cf. figure 2). Tout d'abord, nous avons extrait les journaux d'événements des apprenants qui ont suivi ce cours. Ensuite, nous avons appliqué le regroupement de traces pour identifier des groupes homogènes d'apprenants. Enfin, nous avons découvert des modèles de processus. Lorsqu'on veut considérer un nouvel apprenant pour ce cours avec son propre style d'apprentissage, notre objectif est de recommander à ce nouvel apprenant un scénario pédagogique basé sur son style d'apprentissage et les modèles de processus des apprenants précédents. Dans notre exemple illustratif, étant donné les journaux d'événements de départ, le résultat du regroupement des traces donne un ensemble de 3 journaux d'événements nommés *Event Log 1*, *Event Log 2* et *Event Log 3*. Nous avons procédé à l'application d'un algorithme de découverte de processus sur chaque journal d'événements afin d'extraire les modèles de processus. Nous avons obtenu trois modèles de processus nommés PM_1, PM_2, et PM_3.

Nous procédons ensuite à la caractérisation de chaque regroupement trouvé vis-à-vis du style d'apprentissage. Par exemple, dans le style d'apprentissage de Kolb, il existe quatre styles : convergent, divergent, assimilant et accommodant. Nous caractérisons chaque modèle de processus en fonction du style d'apprentissage de Kolb. Ainsi, après la caractérisation des modèles de processus, nous obtenons LS_1, LS_2 et LS_3 pour PM_1, PM_2 et PM_3. Le nouvel apprenant a un style d'apprentissage N_{LS}. Nous calculons la distance entre N_{LS} et LS_1, LS_2 et LS_3. Nous obtenons la distance la plus proche entre N_{LS} et LS_2, puis LS_3 et LS_1. Par conséquent, nous recommandons PM_2, PM_3 puis PM_1.

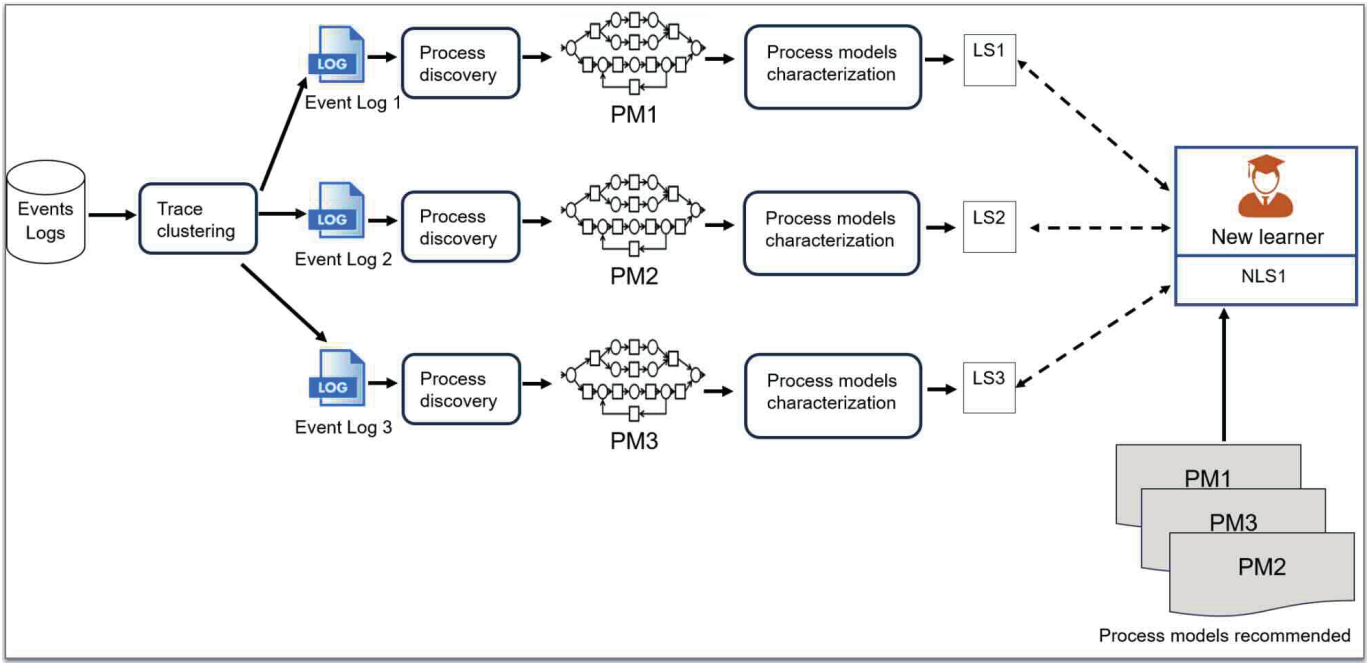

FIGURE 2. *Exemple de scénario de recommandation basée sur les modèles de processus*

Nous utilisons l'IDE PyCharm[1] 3.9 pour notre implémentation. Nous utilisons également la librairie PM4Py [Berti et al., 2019] qui contient les algorithmes classiques pour la découverte de

1. https ://www.jetbrains.com/pycharm/

modèles de processus tels que α-miner, Inductive miner et Heuristic miner. La librairie Python *SentenceTransformers* nous permet de générer des *embeddings* de textes et d'images, issus de *Sentence-BERT* [Reimers & Gurevych, 2019]. Cette librairie offre une large collection de modèles pré-entraînés adaptés à différentes tâches.

5.1. *Jeu de données utilisé*

Nous avons évalué notre approche sur des log réels. Les données utilisées sont téléchargées directement depuis la plateforme Moodle sous la forme d'un fichier CSV. Ce fichier est appelé journal des événements et contient les enregistrements des événements qui se sont produits. Le journal des événements téléchargé contient neuf attributs : heure, nom complet, utilisateur concerné, contexte de l'événement, composant, nom de l'événement, description, origine et adresse IP. Afin d'appliquer l'algorithme de découverte de processus, nous avons utilisé les trois attributs : nom complet, nom de l'événement et heure.

La figure 3 montre un extrait de ce journal d'événements. Dans cette figure, les noms des étudiants sont remplacés par« Utilisateur+numéro » pour garantir leur anonymat.

Time	User full name	Affected user	Event context	Component	Event name	Descripti	Origin	IP address
15/06/23, 12:58	User1	-	Course: BUT-I	System	Course viewed	The user	web	10.113.97.193
15/06/23, 12:57	User1	-	Assignment: D	Assignment	The status of the submission has been viewed.	The user	web	10.113.97.193
15/06/23, 12:57	User1	User1	Assignment: D	Assignment	Feedback viewed	The user	web	10.113.97.193
15/06/23, 12:57	User1	-	Assignment: D	Assignment	Course module viewed	The user	web	10.113.97.193
15/06/23, 12:57	User1	-	Course: BUT-I	System	Course viewed	The user	web	10.113.97.193
15/06/23, 11:41	User2	-	File: sujetDS-2	File	Course module viewed	The user	web	10.113.75.114
15/06/23, 11:40	User2	-	Course: BUT-I	System	Course viewed	The user	web	10.113.75.114
15/06/23, 11:34	User3	-	Assignment: D	Assignment	The status of the submission has been viewed.	The user	web	10.113.71.13
15/06/23, 11:34	User3	User3	Assignment: D	Assignment	Feedback viewed	The user	web	10.113.71.13
15/06/23, 11:34	User3	-	Assignment: D	Assignment	Course module viewed	The user	web	10.113.71.13
15/06/23, 11:34	User3	-	Course: BUT-I	Quiz	Course module instance list viewed	The user	web	10.113.71.13
15/06/23, 10:45	User4	-	Assignment: D	Assignment	The status of the submission has been viewed.	The user	web	10.113.78.41
15/06/23, 10:45	User4	User4	Assignment: D	Assignment	Feedback viewed	The user	web	10.113.78.41
15/06/23, 10:45	User4	-	Assignment: D	Assignment	Course module viewed	The user	web	10.113.78.41
15/06/23, 10:45	User4	-	Course: BUT-I	System	Course viewed	The user	web	10.113.78.41
15/06/23, 10:44	User5	-	Assignment: D	Assignment	The status of the submission has been viewed.	The user	web	92.184.
15/06/23, 10:44	User5	User5	Assignment: D	Assignment	Feedback viewed	The user	web	92.184.
15/06/23, 10:44	User5	-	Assignment: D	Assignment	Course module viewed	The user	web	92.184.
15/06/23, 10:44	User5	-	Course: BUT-I	System	Course viewed	The user	web	92.184.

FIGURE 3. *Extrait d'un journal d'événement Moodle*

Comme indiqué précédemment, la couche de recommandation est basée sur le style d'apprentissage de l'apprenant. C'est pourquoi nous avons demandé aux apprenants de répondre à un questionnaire en ligne qui génère le style d'apprentissage de l'apprenant sur la base du style d'apprentissage de Kolb. Une fois que l'étudiant a rempli le questionnaire, ses préférences en matière de style d'apprentissage sont enregistrées afin d'évaluer la couche de recommandation. La figure 4 montre un extrait des styles d'apprentissage de Kolb des apprenants. Les noms des étudiants sont cachés afin de préserver leur anonymat. Chaque apprenant est caractérisé par quatre valeurs correspondant à un des styles d'apprentissage de Kolb. Par exemple, le premier apprenant de cette figure a une valeur de 1 pour « accommodant », 0 pour « divergent », 1 pour « convergent » et 0 pour « assimilant ».

5.2. *Évaluation de la couche fouille de processus*

Pour évaluer la couche fouille de processus, nous avons réalisé des expérimentations sur les deux jeux de données IHM_1 et IHM_2. La spécificité du domaine de l'enseignement rend difficile la col-

Learners	Accommodating	Diverging	Converging	Assimilating
[redacted]	1	0	1	0
[redacted]	0	1	0	0
[redacted]	1	0	1	0
[redacted]	1	0	1	0
[redacted]	0	1	0	1
[redacted]	1	0	1	0
[redacted]	0	1	0	0
[redacted]	0	1	0	0
[redacted]	1	1	1	0

FIGURE 4. *Extrait du classement des styles d'apprentissage de Kolb des étudiants*

lecte de traces importantes (la taille est liée à la taille des cohortes) et la variabilité des contenus fait qu'il n'est pas possible de les regrouper. Pour chaque jeu de données : nous avons d'abord appliqué un algorithme de fouille de processus afin d'avoir des mesures sans *clustering* ; puis nous avons utilisé différents algorithmes de *clustering* tels que k-means, DBSCAN [Ester et al., 1996], Agglomerative Clustering [Müllner, 2011] et Gaussian Mixture Model (GMM)[2] ; nous avons ensuite appliqué l'algorithme *heuristic miner* pour découvrir le modèle de processus correspondant à chaque *cluster* (cet algorithme a été choisi car il a donné de meilleurs résultats dans nos travaux précédents [Hachicha et al., 2021], un exemple est donné par la figure 5 pour le cluster C_2) ; enfin, nous avons utilisé les mesures d'évaluation Fitness (F), Précision (P) et Généralisation (G) pour évaluer les algorithmes de *clustering* à travers la qualité des modèles de processus découverts.

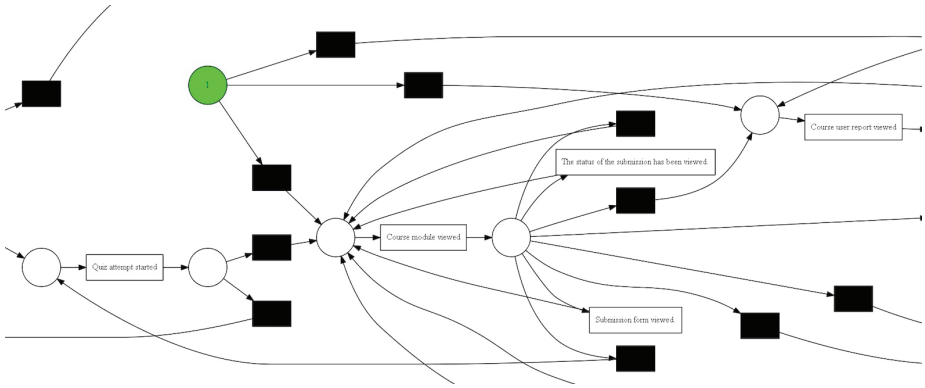

FIGURE 5. *Extrait du modèle de parcours d'apprentissage découvert pour le cluster* C_2

Les résultats des expérimentations sont données dans les tableaux 2 et 3. Nous avons déterminé que le nombre de *clusters* serait de 5 après une investigation expérimentale. Ensuite, nous avons calculé la moyenne de chaque métrique (Fitness, Précision et Généralisation) afin d'identifier l'algorithme de *clustering* le plus performant. Les *clusters* sont nommés C_0, C_1, C_2, C_3 et C_4. La moyenne de chaque métrique mesurée est représentée par l'abréviation Avg.

Pour le jeu de données IHM_1 les résultats montrent que les modèles de processus découverts à partir des *clusters* générés par l'algorithme *Gaussian Mixture Model* (GMM) sont meilleurs que ceux découverts avec les autres algorithmes [Hachicha et al., 2022]. Nous avons obtenu une valeur de *Fitness* égale à 0.9837, ce qui signifie que le modèle représente mieux les comportements présents dans le journal des événements. De plus, il possède la valeur la plus élevée de Précision (0.3489), indiquant que le modèle de processus détecte correctement 34.89% des activités réelles présentes dans le journal des événements

2. « User Guide. » Gaussian mixture models. Web. © 2007 - 2022. scikit-learn developers.

	k-means			DBSCAN			Agglomerative			GMM		
	F	P	G	F	P	G	F	P	G	F	P	G
C_0	0.9922	0.1951	0.7718	0.9887	0.1441	0.7852	0.9892	0.2037	0.8019	0.9862	0.2393	0.8429
C_1	0.9847	0.2218	0.8235	0.9841	0.2479	0.8554	0.9796	0.2960	0.7447	0.9799	0.1972	0.7597
C_2	0.9652	0.1624	0.6668	0.9865	0.1909	0.6172	0.9923	0.1485	0.7963	0.9652	0.1624	0.6668
C_3	0.9838	0.2884	0.7663	0.9332	1	0.7836	0.9652	0.1624	0.6668	0.9923	0.1459	0.7607
C_4	0.9923	0.1486	0.7938	0.9802	0.1586	0.6003	0.9909	0.1623	0.7553	0.9949	1	0.8162
Avg	0.9836	0.2023	0.7644	0.9745	0.3483	0.7283	0.9834	0.1945	0.7530	**0.9837**	**0.3489**	**0.7692**

TABLEAU 2. *Mesures de qualité pour les différents algorithmes de clustering sur les données* IHM_1

	k-means			DBSCAN			Agglomerative			GMM		
	F	P	G	F	P	G	F	P	G	F	P	G
C_0	0.9859	0.2336	0.6790	0.9915	0.1563	0.7481	0.9764	0.36	0.6657	0.9905	0.2102	0.7269
C_1	0.9807	0.1986	0.5946	0.8688	1.0	0.7371	0.9816	0.1976	0.6385	0.9863	0.2393	0.7157
C_2	0.9905	0.2102	0.7269	0.8583	0.1999	0.7389	0.9905	0.2102	0.7269	0.9807	0.1986	0.5946
C_3	0.9764	0.36	0.6657	0.0540	0.0616	0.5028	0.9859	0.2336	0.6790	0.0264	0.0540	0.5082
C_4	0.9578	0.2007	0.5899	0.9872	1.0	0.8100	0.9577	0.2448	0.6384	0.9578	0.2007	0.5899
Avg	0.9782	0.2406	0.6512	0.7519	0.4835	0.7073	**0.9784**	**0.2492**	**0.6697**	0.7883	0.1805	0.6270

TABLEAU 3. *Mesures de qualité pour les différents algorithmes de clustering sur les données* IHM_2

par rapport à toutes les activités qu'il a identifiées. Enfin, la Généralisation est de 0.7692, confirmant que le modèle est capable de généraliser le comportement présent dans le journal des événements.

En ce qui concerne le jeu de données IHM_2, nous constatons que les modèles de processus découverts sur la base des *clusters* générés par l'*Agglomerative Clustering* se sont révélés être les meilleurs par rapport aux autres algorithmes.

Pour valider l'apport du *trace clustering*, nous avons également comparé les modèles de processus découverts sans regroupement de traces. Nous obtenons les valeurs $F = 0.9903$, $P = 0.1596$ et $G = 0.7611$ pour le jeu de données IHM_1. Ces résultats montrent que le *trace clustering* n'altère pas la qualité des modèles de processus découverts, mais au contraire, l'améliore. Cela prouve aussi que le *trace clustering* a un impact positif sur les modèles de processus issus des jeux de données du domaine de l'éducation et dépourvus d'informations supplémentaires sur les apprenants (résultats d'apprentissage et caractéristiques des apprenants).

Nous pouvons noter que la valeur de la Précision a augmenté de 0.1596 à 0.3489 et la valeur de Généralisation a augmenté relativement. Cependant, la valeur de *Fitness* a légèrement diminué. Cette diminution est due au fait que le *trace clustering* crée des groupes plus homogènes de traces, ce qui peut réduire légèrement la capacité des modèles de processus à représenter toute la diversité des données initiales contenues dans le journal d'événements. Cependant, cette diminution est généralement compensée par une meilleure précision et généralisation des modèles pour chaque groupe spécifique.

5.3. *Évaluation de la couche recommandation*

Pour valider l'algorithme de recommandation des modèles de processus proposé, nous avons mené deux phases de validation distinctes visant à évaluer l'impact des transformeurs et du *trace clustering* sur la recommandation des modèles de processus. Pendant ces deux phases, nous utilisons les mesures d'évaluation des systèmes de recommandation MAE et RMSE.

Ces deux mesures (l'erreur absolue moyenne - MAE, et l'erreur quadratique moyenne - RMSE) sont calculées selon [Willmott & Matsuura, 2005]. Plus les valeurs de MAE et RMSE sont faibles, plus la précision de la recommandation est élevée. Il s'agit des mesures les plus couramment utilisées pour mesurer la précision des systèmes de recommandation. La MAE (cf. équation (5.7)) représente l'écart absolu entre les modèles de processus prédits (p_i) et les modèles de processus réels classés (r_i).

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|p_i - r_i| \quad [5.7]$$

Le RMSE (cf. équation (5.8)) est l'écart type des erreurs de prédiction.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - r_i)^2} \quad [5.8]$$

5.3.1. Apport des transformeurs

Afin de calculer la similarité sémantique entre les descriptions des styles d'apprentissage et les noms de toutes les activités, nous avons utilisé des transformeurs de phrases. En effet, pour cette phase, nous avons employé le *package* Python *SentenceTransformers* qui est capable de générer des *embeddings* (texte, image, etc.), basés sur *Sentence-BERT* [Reimers & Gurevych, 2019]. *SentenceTransformers* propose une large collection de modèles pré-entraînés adaptés à diverses tâches.

Afin de déterminer le meilleur transformeur, nous avons effectué plusieurs expérimentations avec différents modèles tels que multi-qa-mpnet-base-dot-v1 (MPNET), multi-qa-distilbert-dot-v1 (DistilBERT), stsb-roberta-large (RoBERTa), all-MiniLM-L6-v2 (MiniLM L6), et all-MiniLM-L12-v2 (MiniLM L12).

Le tableau 4 affiche les valeurs de MAE et RMSE obtenues pour certains apprenants L_1, L_2, L_3, L_4 et L_5 en tenant compte de la fréquence lors du calcul des valeurs de caractérisation des modèles de processus.

	MPNET		DistrilBERT		RoBERTa		MiniLM L6		MiniLM L12	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
L1	1.6	1.7888	2	2.6076	1.2	1.6733	2	2.6076	2	2.6076
L2	0.8	0.8944	0.8	0.8944	0.8	0.8944	1.6	2.0976	1.2	1.6733
L3	2	2.1908	2	2.1908	1.2	1.5491	2	2.1908	2	2.1908
L4	0.4	0.6324	2	2.6076	1.2	1.5491	1.2	1.5491	1.2	1.6733
L5	1.6	2.2803	0.4	0.6324	1.6	1.8973	2.4	2.5298	2.4	2.5298
Avg	1.28	1.5573	1.44	1.7865	1.2	1.5126	1.84	2.1949	1.76	2.1349

TABLEAU 4. *Mesures d'évaluation en fonction des différents modèles de SentenceTransformers*

Nous avons obtenu les meilleurs résultats avec le modèle stsb-roberta-large (RoBERTa). Ces résultats indiquent que la marge de MAE pour ce modèle est de 1.2, ce qui signifie que les prédictions du modèle sont en moyenne écartées de 1.2 unités de la valeur réelle, et que la dispersion des erreurs prédites par ce modèle est relativement faible, avec une valeur de RMSE de 1.5126. En d'autres termes, cela met en évidence la performance de RoBERTa par rapport aux autres modèles évalués.

5.3.2. Apport du *Trace Clustering*

Après avoir validé l'efficacité des transformeurs, notamment du modèle RoBERTa, pour caractériser les styles d'apprentissage des modèles de processus et son impact sur les résultats de recommandation, nous cherchons à valider l'impact du *trace clustering* sur les résultats de la recommandation. Rappelons que la caractérisation d'un modèle de processus en fonction des styles d'apprentissage repose non seulement sur les transformeurs, mais aussi sur les fréquences des activités qui le composent (cf. formule 4.6). Ces fréquences sont extraites des modèles de processus obtenus après l'application de l'algorithme de *trace clustering*, utilisé pour découvrir les modèles de processus.

Dans cette optique, nous avons mené deux expérimentations, l'une prenant en considération les fréquences des activités et l'autre les ignorant.

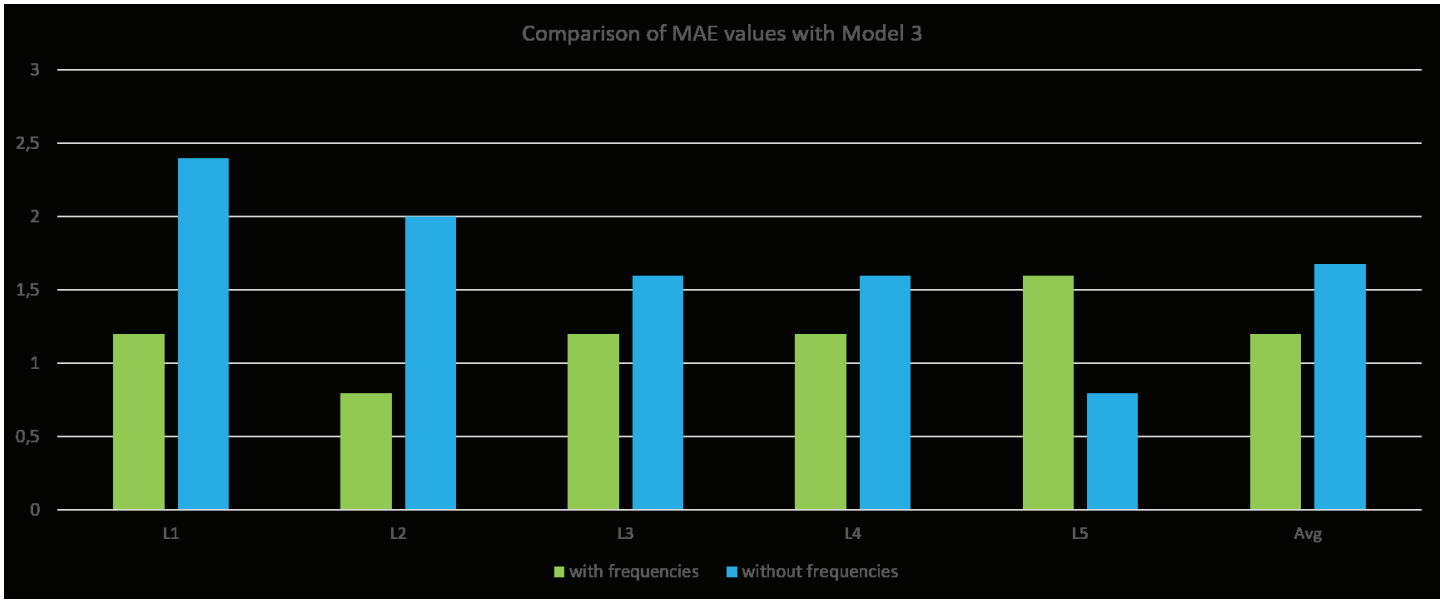

FIGURE 6. *Comparaison des valeurs de MAE pour le modèle RoBERTa*

FIGURE 7. *Comparaison des valeurs de RMSE pour le modèle RoBERTa*

Les figures 6 et 7 montrent les résultats de validation pour cinq apprenants utilisant le modèle RoBERTa. La valeur moyenne de MAE est améliorée par la plus basse valeur qui est égale à 1.2 lors de

l'inclusion des fréquences d'activités, comparée à 1.68 sans cette considération. De même, la valeur moyenne de RMSE est de 1.5126 en prenant en compte les fréquences des activités, tandis qu'il atteint 1.9536 sans cette considération. Ces résultats soulignent l'importance de considérer les fréquences des activités lors de la caractérisation des styles d'apprentissage des modèles de processus, démontrant ainsi l'impact positif du *trace clustering* non seulement dans la découverte des modèles de processus, mais également dans la recommandation.

6. Conclusion

Dans le cadre de l'élaboration d'un outil d'aide à la construction d'un parcours personnalisé, nous visons à développer un outil qui recommande en se basant sur la fouille de processus. Nous avons présenté le domaine de la fouille de processus ainsi que les principes de la recommandation. Puis nous avons proposé un état de l'art des travaux utilisant la fouille de processus pour l'apprentissage. Nous n'avons pas trouvé de travaux utilisant le *trace clustering* pour la recommandation dans le domaine de la formation.

Le *trace clustering* vise à regrouper des traces d'exécutions qui possèdent des dynamiques proches. Nous avons proposé une architecture pour la recommandation qui utilise le *feature based clustering*. Nous avons validé cette architecture à partir de données issues d'un cours d'introduction à la programmation d'IHM comportant 42438 événements. Nous avons montré que nous pouvons déterminer des regroupements pertinents. Nous avons ensuite caractérisé chaque regroupement en fonction de son style d'apprentissage, puis formulé une recommandation sur un parcours pour un nouvelle apprenant.

L'intérêt de passer par le *trace clustering* est que l'on détermine à quelle catégorie de parcours d'apprentissge un apprenant se rattache. Pour ce parcours nous pouvons extraire un modèle et ainsi nous pouvons fournir des éléments d'explication de la proposition de recommandation faite.

Ces travaux constituent une première étape afin d'améliorer l'efficacité du processus de suivi et d'anticipation des actions de l'utilisateur en lien avec un objectif déclaré ou estimé. Nous visons à développer une plateforme numérique définissant un espace facilitateur et fédérateur d'engagement installant une dynamique nouvelle de l'occupation de l'espace d'apprentissage et d'enseignement et intégrant la gestion personnalisée ou collaborative des ressources d'apprentissage. Nous visons à favoriser l'accessibilité, la continuité et la porosité de l'apprentissage en s'appuyant sur la recommandation des ressources adaptées à un projet personnel ou professionnel d'apprentissage réalisés sur un territoire.

La méthode que nous proposons, qui consiste à considérer le parcours d'un apprennant comme un processus, puis à utiliser les traces laissées par l'apprenant pour regrouper et découvrir les modèles de comportement pour recommander la prochaine action afin d'atteindre un objectif, peut être utilisée dans différents contextes. Par exemple celui des bibliothèques numériques où il est possible de caractériser la navigation de l'utilisateur afin d'adapter le contenu proposé.

Ce besoin de caratériser la navigation de l'utilisateur se retrouve également au niveau des sites web. Nous nous rapprochons alors du *Web Usage Mining*, cependant nous proposons un modèle de navigation qui permet de construire un raisonnement explicable pour la recommandation. Notre approche trouverait tout son intérêt pour des environnements où l'utilisateur déroule son propre processus (banque, assurance, administration...).

Remerciements

Ce travail a été soutenu financièrement par le programme PHC Utique du Ministère français des Affaires étrangères et du Ministère de l'Enseignement supérieur et de la recherche et par le Ministère tunisien de l'Enseignement supérieur et de la recherche scientifique dans le cadre du projet CMCU numéro 22G1403.

Références

AALST, W. Van der. (2016). *Process Mining : Data Science in Action*. Springer.

AALST, W. Van der, WEIJTERS, A., & MĂRUŞTER, L. (2004). *Workflow Mining : Discovering Process Models from Event Logs*. IEEE Transactions on Knowledge and Data Engineering, 16(9), 1128–1142.

ABDELGHANI, W., ZAYANI, C. A., AMOUS, I., & SÈDES, F. (2018). *Trust evaluation model for attack detection in social internet of things*. In *13th International Conference Risks and Security of Internet and Systems : CRiSIS'2018* (pp. 48–64). Arcachon, France.

ADRIANSYAH, A. (2014). *Aligning observed and modeled behavior*. Ph.D. thesis, Technische Universiteit Eindhoven.

AGRAWAL, R., GUNOPULOS, D., & LEYMANN, F. (1998). *Mining process models from workflow logs*. In H. J. Schek, G. Alonso, F. Saltor, & I. Ramos (Eds.), *Advances in Database Technology — EDBT'98* (pp. 467–483). Springer Berlin Heidelberg.

AL FARARNI, K., AGHOUTANE, B., RIFFI, J., SABRI, A., & YAHYAOUY, A. (2020). *Comparative study on approaches of recommendation systems*. In *Embedded Systems and Artificial Intelligence : Proceedings of ESAI'2019, Fez, Morocco* (pp. 753–764).

BEEMT, A. van den, BUIJS, J., & VAN DER AALST, W. (2018). *Analysing Structured Learning Behaviour in Massive Open Online Courses (MOOCs) : An Approach Based on Process Mining and Clustering*. *International Review of Research in Open and Distributed Learning, 19*(5), 37–60.

BERLAND, M., BAKER, R., & BLIKSTEIN, P. (2014). *Educational Data Mining and Learning Analytics : Applications to Constructionist Research*. *Technology, Knowledge and Learning, 19*, 205–220.

BERTI, A., VAN ZELST, S. J., & AALST, W. van der. (2019). *Process mining for python (PM4Py) : bridging the gap between process-and data science*. *arXiv preprint arXiv :1905.06169*.

BOBADILLA, J., ORTEGA, F., HERNANDO, A., & GUTIÉRREZ, A. (2013). *Recommender systems survey*. *Knowledge-based systems, 46*, 109–132.

BUIJS, J. C., VAN DONGEN, B. F., & VAN DER AALST, W. M. (2012, September). *On the role of fitness, precision, generalization and simplicity in process discovery*. In *On the Move to Meaningful Internet Systems (OTM'2012) Confederated International Conferences* (pp. 305–322). Rome, Italy.

CAMPANA, M. G., & DELMASTRO, F. (2017). *Recommender systems for online and mobile social networks : A survey*. *Online Social Networks and Media, 3*, 75–97.

Çano, E., & MORISIO, M. (2017). *Hybrid recommender systems : A systematic literature review*. *Intelligent data analysis, 21*(6), 1487–1524.

CHANDRASEKARAN, D., & MAGO, V. (2021). *Evolution of semantic similarity—a survey*. *ACM Computing Surveys (CSUR), 54*(2), 1–37.

COOK, J. E., & WOLF, A. L. (1998). *Discovering models of software processes from event-based data*. *ACM Trans. Softw. Eng. Methodol., 7*, 215-249.

CORDIER, A., LEFEVRE, M., CHAMPIN, P. A., GEORGEON, O., & MILLE, A. (2013). *Trace-Based Reasoning - Modeling interaction traces for reasoning on experiences*. In *26th International FLAIRS Conference*, St. Pete Beach, US. (pp. 1-15).

DEVLIN, J., CHANG, M. W., LEE, K., & TOUTANOVA, K. (2018). *Bert : Pre-training of deep bidirectional transformers for language understanding*. *arXiv preprint arXiv :1810.04805*.

DIAMANTINI, C., GENGA, L., & POTENA, D. (2016). *Behavioral process mining for unstructured processes*. *Journal of Intelligent Information Systems, 47*(1), 5–32.

DWIVEDI, P., KANT, V., & BHARADWAJ, K. K. (2018). *Learning path recommendation based on modified variable length genetic algorithm*. *Education and information technologies, 23*, 819–836.

ESTER, M., KRIEGEL, H. P., SANDER, J., XU, X., et al. (1996). *A density-based algorithm for discovering clusters in large spatial databases with noise*. In *2nd Int. Conf. on Knowledge Discovery and Data Mining* (pp. 226–231).

FELDER, R., & SILVERMAN, L. (1988). *Learning and Teaching Styles in Engineering Education. Journal of Engineering Education, 78*(7), 674-681.

GHORBEL, L., ZAYANI, C., & AMOUS, I. (2015). *Improve the Adaptation Navigation in Educational Cross-systems. Procedia Computer Science, 60*, 662-670.

HACHICHA W., GHORBEL L., CHAMPAGNAT R., RABAH M., NOWAKOWSKI S., ZAYANI C. A. (2023). *Proposition d'une architecture utilisant le trace clustering pour recommander un parcours d'apprentissage*. In C. Ponsard, C. Faucher (Eds.), Actes du 41e congrès inforsid, p. 149–164. La Rochelle, France.

HACHICHA, W., GHORBEL, L., CHAMPAGNAT, R., ZAYANI, C. A., & AMOUS, I. (2021). *Using Process Mining for Learning Resource Recommendation : A Moodle Case Study. Procedia Computer Science, 192*, 853-862. Knowledge-Based and Intelligent Information & Engineering Systems : Proceedings of the 25th International Conference KES2021

HACHICHA, W., GHORBEL, L., CHAMPAGNAT, R., & ZAYANI, C. A. (2022). *Trace Clustering Based on Activity Profile for Process Discovery in Education*. In *22th Int. Conf. on Intelligent Systems Design and Applications (ISDA 2022)* (pp. 545–554).

HO, H. N., RABAH, M., NOWAKOWSKI, S., & ESTRAILLIER, P. (2016). *Toward a Trace-Based PROMETHEE II Method to answer " What can teachers do ? " in Online Distance Learning Applications*. In *13th International Conference on Intelligent Tutoring Systems* (pp. 480-484). Zagreb, Croatia

KHANAL, S. S., PRASAD, P., ALSADOON, A., & MAAG, A. (2020). *A systematic review : machine learning based recommendation systems for e-learning. Education and Information Technologies, 25*, 2635–2664.

KODAMA, K., IIJIMA, Y., GUO, X., & ISHIKAWA, Y. (2009). *Skyline queries based on user locations and preferences for making location-based recommendations*. In *Proceedings of the 2009 International Workshop on Location Based Social Networks* (pp. 9–16).

KOLB, D. A., & OTHERS. (2007). *The Kolb learning style inventory. Hay Resources Direct Boston*.

KOLEKAR, S. V., PAI, R. M., & MM M P. (2019). *Rule based adaptive user interface for adaptive E-learning system. Education and Information Technologies, 24*, 613–641.

LEBLAY, J., RABAH, M., CHAMPAGNAT, R., & NOWAKOWSKI, S. (2018). *Process-based Assistance Method for Learner Academic Achievement*. In *E-Learning Conference (EL'2018)* (pp. 89-96).

LI, G., & DE CARVALHO, R. M. (2019). *Process Mining in Social Media : Applying Object-Centric Behavioral Constraint Models. IEEE Access, 7*, 84360–84373.

LIU, Y., OTT, M., GOYAL, N., DU, J., JOSHI, M., & CHEN, D. (2019). *Roberta : A robustly optimized bert pretraining approach. arXiv preprint arXiv :1907.11692*.

MARTINEZ, P., MONTAÑES, O., SERRALTA, J. M., & TANSINI, L. (2021). *Modelling Computer Engineering Student Trajectories with Process Mining*. In *Latin American Conference on Learning Analytics* (pp. 48–57).

MEZGHANI, M., PÉNINOU, A., ZAYANI, C. A., AMOUS, I., & SÈDES, F. (2017). *Producing relevant interests from social networks by mining users' tagging behaviour : A first step towards adapting social information. Data & Knowledge Engineering, 108*, 15–29.

MÜLLNER, D. (2011). *Modern hierarchical, agglomerative clustering algorithms. arXiv preprint arXiv :1109.2378*.

MYERS, I. B. (1985). *A Guide to the Development and Use of the Myers-Briggs Type Indicator : Manual. Consulting Psychologists Press*.

NAFEA, S. M., SIEWE, F., & HE, Y. (2019). *On recommendation of learning objects using felder-silverman learning style model. IEEE Access, 7*, 163034–163048.

PIKA, A., WYNN, M. T., BUDIONO, S., HOFSTEDE, A. H. ter, AALST, W. M. van der, & REIJERS, H. A. (2019). *Towards Privacy-Preserving Process Mining in Healthcare*. In *Business Process Management Workshops* (pp. 483–495). *Springer International Publishing*.

REAL, E. M., PIMENTEL, E. P., & BRAGA, J. C. (2021). *Analysis of Learning Behavior in a Programming Course using Process Mining and Sequential Pattern Mining*. In *2021 IEEE Frontiers in Education Conference (FIE)* (pp. 1-9).

REAL, E. M., PIMENTEL, E. P., OLIVEIRA, L. V. de, BRAGA, J. C., & STIUBIENER, I. (2020). *Educational process mining for verifying student learning paths in an introductory programming course*. In *2020 IEEE Frontiers in Education Conference (FIE)* (pp. 1–9).

REIMERS, N., & GUREVYCH, I. (2019). *Sentence-BERT : Sentence Embeddings using Siamese BERT-Networks. CoRR, abs/1908.10084*. http://arxiv.org/abs/1908.10084

ROMERO, C., & VENTURA, S. (2013). *Data Mining in Education. Wiley Int. Rev. Data Min. and Knowl. Disc., 3*(1), 12–27.

ROMERO, C., VENTURA, S., & GARCÍA, E. (2008). *Data mining in course management systems : Moodle case study and tutorial. Computers & Education, 51*(1), 368–384.

SANH, V., DEBUT, L., CHAUMOND, J., & WOLF, T. (2019). *DistilBERT, a distilled version of BERT : smaller, faster, cheaper and lighter*. *arXiv preprint arXiv :1910.01108*.

SEDRAKYAN, G., SNOECK, M., & DE WEERDT, J. (2014). *Process mining analysis of conceptual modeling behavior of novices – empirical study using JMermaid modeling and experimental logging environment*. Computers in Human Behavior, **41**(486-503).

SEIDEL, N., RIEGER, C M., & WALLE, T. (2020). *Semantic Textual Similarity of Course Materials at a Distance-Learning University*. In Proceedings of 4th Educational Data Mining in Computer Science Education (CSEDM) Workshop co-located with the 13th Educational Data Mining Conference (EDM 2020).

SHOKEEN, J., & RANA, C. (2020). *A study on features of social recommender systems*. Artificial Intelligence Review, 53(965–988).

SONG, M., GÜNTHER, C W., & AALST, W M. Van der (2008). *Trace clustering in process mining*. In International Conference on Business Process Management (pp. 109–120).

TANG, J., HU, X., & LIU, H. (2013). *Social recommendation : a review*. Social Network Analysis and Mining, 3, 1113–1133.

TRABELSI, M., SUIRE, C., MORCOS, J., & CHAMPAGNAT, R. (2019). *Fouille de processus auto-définis : cas d'étude d'un moteur de recherche d'une bibliothèque numérique*. In Actes du 37e Congrès INFORSID (pp. 131–146).

TRABELSI, M., SUIRE, C., MORCOS, J., & CHAMPAGNAT, R. (2019). *User's Behavior in Digital Libraries : Process Mining Exploration*. In A. Doucet, A. Isaac, K. Golub, T. Aalberg & A. Jatowt (*Eds.*), Digital Libraries for Open Knowledge (pp. 388–392). Springer International Publishing.

TRABELSI, M., SUIRE, C., MORCOS, J., & CHAMPAGNAT, R. (2021). *A New Methodology to Bring Out Typical Users Interactions in Digital Libraries*. In 2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL) (pp. 11–20).

TROUDI, A., GHORBEL, L., AMEL ZAYANI, C., JAMOUSSI, S., & AMOUS, I. (2020). *MDER : Multi-Dimensional Event Recommendation in Social Media Context*. The Computer Journal, 64(3), 369–382.

TRUONG, H M. (2016). *Integrating learning styles and adaptive e-learning system : Current developments, problems and opportunities*. Computers in human behavior, 55, 1185–1193.

WEIJTERS, A., & RIBEIRO, J. (2011). *Flexible heuristics miner (FHM)*. In Computational Intelligence and Data Mining (pp. 310–317).

WILLMOTT, C J., & MATSUURA, K. (2005). *Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance*. Climate research, 30(1), 79–82.

WOLF, T., DEBUT, L., SANH, V., CHAUMOND, J., DELANGUE, C., & MOI, A. et al. (2020). *Transformers : State-of-the-art natural language processing*. In Proceedings of the 2020 conference on empirical methods in natural language processing : system demonstrations (pp. 38–45).

YAN, L., YIN, C., CHEN, H., RONG, W., XIONG, Z., & DAVID, B. (2021). *Learning Resource Recommendation in E-Learning Systems Based on Online Learning Style*. In 14th International Conference on Knowledge Science, Engineering and Management, KSEM 2021 (pp. 373–385). Tokyo, Japan.

ZANDKARIMI, F., REHSE, J R., SOUDMAND, P., & HOEHLE, H. (2020). *A Generic Framework for Trace Clustering in Process Mining*. In 2020 2nd International Conference on Process Mining (pp. 177–184).

ZHANG, S., ZHENG, X., & HU, C. (2015). *A survey of semantic similarity and its application to social network analysis*. In 2015 IEEE International Conference on Big Data (Big Data) (pp. 2362–2367).

