Alignement Sémantique d'Apprentissages Critiques par *Embeddings* Contextuels

Wissal Essalah[1], Nicolas Delestre[2], Nicolas Malandain[2] et Damien Olivier[1]

[1] LITIS, Université Le Havre Normandie, Normandie Université, France
[2] LITIS, INSA Rouen Normandie, Normandie Université, France
wissal.essalah@etu.univ-lehavre.fr

Résumé. Nous étudions l'alignement entre Apprentissages Critiques (AC) et Attendus d'Apprentissage Disciplinaire (AAD), enjeu majeur de l'approche par compétences. Le problème est formulé comme une tâche de recherche d'information : les AC sont vus comme des requêtes, les AAD comme des documents classés par similarité sémantique. Nous comparons des modèles d'embeddings statiques (Word2Vec, FastText) et dynamiques (`SBERT`, `sentence-camembert-large`). L'analyse des courbes ROC et score AUC montrent sur nos jeux de données des résultats de moyens à bons.

Mots-clé: Approche par compétences, Évaluation pédagogique, Apprentissages Critiques, Attendus d'Apprentissage Disciplinaire, Recherche d'information, Alignement sémantique, *Embeddings* statiques, *Embeddings* dynamiques

Abstract. We study the alignment between Critical Learning Outcomes (AC) and Disciplinary Learning Objectives (AAD), a key issue in competency-based education. The task is framed as an information retrieval problem : ACs are considered as queries and AADs as documents ranked by semantic similarity. We compare static (Word2Vec, FastText) and contextual (`SBERT`, `sentence-camembert-large`) embedding models. The analysis of ROC curves and AUC scores shows results ranging from average to good on our datasets.

Keywords: Competency-based approach, Educational assessment, Critical Learning Outcomes, Disciplinary Learning Objectives, Information retrieval, Semantic alignment, Static embeddings, Contextual embeddings

1 Introduction

Depuis quelques années, les organismes de tutelle des établissements d'enseignement supérieurs français incitent à passer d'une approche programme [1] à une approche par compétences (APC) [4]. Ce changement de paradigme s'inscrit dans une dynamique de renforcement des compétences clés des étudiants.

Lors des évaluations, et en jury, les enseignants doivent alors vérifier que ces compétences et leurs prérequis sont partiellement ou totalement acquis. Ces compétences, nommées aussi savoir-agir [16], sont évaluées par des projets intégratifs. Les prérequis (savoir, savoir-faire et savoir-être), nommés « apprentissages critiques » (AC), sont enseignés et évalués dans le cadre des cours. L'évaluation de ces AC pose problème car d'une part ils sont transversaux aux différentes disciplines, aux différents cours, et d'autre part ces prérequis sont définis de manière abstraite non opérationnelle. Une façon d'évaluer ces AC est de prendre en compte leurs instanciations opérationnelles, que l'on nomme ici « Attendus d'Apprentissage Disciplinaire » (AAD), au sein des cours. Dans la formation ITI de l'INSA Rouen Normandie, le tableau 1 présente deux exemples d'AC avec quelques AAD d'un cours d'algorithmique pour la compétence « spécifier ». Cela signifie que lorsqu'un étudiant valide l'AAD « Décomposer logiquement un problème », il valide en partie l'AC « Décomposer et organiser un processus ».

Apprentissages Critiques	**Attendus d'Apprentissage Disciplinaire**
Décomposer et organiser un processus	AAD1. Formaliser les préconditions d'une opération d'un Type Abstrait de Données AAD2. Définir des Type Abstrait de Données génériques AAD3. Décomposer logiquement un problème
Utiliser un langage formel	AAD1. Formaliser les préconditions d'une opération d'un Type Abstrait de Données AAD4. Formaliser des opérations d'un Type Abstrait de Données AAD5. Formaliser des axiomes ou savoir définir la sémantique d'une opération d'un Type Abstrait de Données

Tableau 1 – Extrait d'alignements d'AC et AAD pour la compétence « spécifier »

Deux problèmes apparaissent alors : l'identification de ces AAD au sein d'un cours et l'alignement des AAD aux AC. Dans cet article, nous traitons du problème de l'alignement des AAD avec les AC en supposant que ces AAD et AC ont été préalablement identifiés.

2 État de l'art

L'alignement des AC et des AAD peut être abordé comme un problème de « recherche d'information » (RI), dans lequel chaque AC joue le rôle de requête et les AAD celui de documents candidats à classer selon leur pertinence [9].

Les AC et les AAD sont transformés en vecteurs numériques appelés *embeddings*, permettant de calculer leur similarité. L'efficacité de cette méthode

dépend de la capacité des modèles d'*embedding* à représenter finement la sémantique des formulations textuelles [17].

2.1 *Embedding*

L'*embedding* est une représentation vectorielle utilisée en TAL pour comparer des mots, des phrases ou des documents sur le plan sémantique [17]. Ces représentations se divisent en deux grandes familles : statiques et dynamiques.

Les *embeddings* statiques associent à chaque mot un vecteur fixe, indépendamment du contexte. Word2Vec [10], FastText [2], GloVe [11] et LexVec [15] en sont des exemples. Au niveau des phrases, la moyenne des vecteurs mots est souvent utilisée. D'autres modèles comme USE [3] produisent des vecteurs de phrases fixes mais insensibles au contexte.

Les *embeddings* dynamiques produisent des vecteurs qui varient selon le contexte d'apparition. ELMo [12], BERT [6], `SBERT` [14] et T5 [13] modélisent finement les dépendances sémantiques et syntaxiques. Pour la langue française, CamemBERT et `sentence-camembert-large` offrent des représentations contextuelles adaptées aux tâches de similarité.

2.2 Mesure de la similarité

La similarité entre représentations vectorielles est fréquemment mesurée à l'aide de la similarité cosinus [8], une métrique largement utilisée en traitement automatique des langues pour comparer des *embeddings*. Elle évalue l'angle entre deux vecteurs u et v selon :

$$\text{Similarité cosinus}(u, v) = \frac{u \cdot v}{\|u\| \, \|v\|}$$

Cette mesure renvoie une valeur comprise entre -1 et 1 : plus elle est proche de 1, plus les deux vecteurs sont considérés comme sémantiquement proches.

2.3 Critères d'évaluation

La qualité du classement produit par un système de recherche d'information est évaluée à l'aide de la courbe ROC (*Receiver Operating Characteristic*) et de l'AUC (*Area Under Curve*) [5]. Une AUC proche de 1 indique que les documents pertinents sont bien classés en tête. Une AUC $> 0{,}9$ est considérée comme excellente, entre 0,8 et 0,9 elle est bonne, entre 0,7 et 0,8 acceptable, et inférieure à 0,7 considérée comme faible [7].

Le tableau 2 montre un exemple de moteur de RI retournant 10 documents, dont 4 pertinents. La figure 1 présente la courbe ROC correspondante avec une AUC de 0.83.

Rang	1	2	3	4	5	6	7	8	9	10
Pertinence	1	1	0	1	0	0	1	0	0	0
x_i	0/6	0/6	1/6	1/6	2/6	3/6	3/6	4/6	5/6	6/6
y_i	1/4	2/4	2/4	3/4	3/4	4/4	4/4	4/4	4/4	4/4

TABLEAU 2 – Résultats jouets d'un moteur de RI

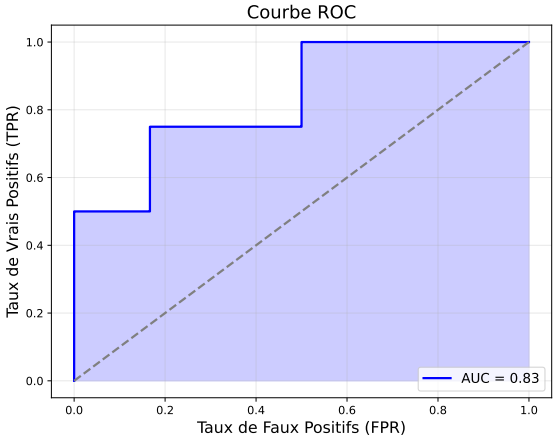

FIG. 1 – Courbe ROC correspondant aux données du tableau 2

3 Protocole expérimental

Pour valider notre approche, nous utilisons les AC définis dans la spécialité ITI de l'INSA Rouen Normandie, structurés autour de six savoir-agir (spécifier, modéliser, communiquer, concevoir, développer, gérer), chacun décomposé en quatre jalons pédagogiques. Les responsables de plusieurs cours ont identifié les AAD et réalisé un alignement avec les AC, constituant une base d'évaluation réaliste. Le tableau 3 résume les principales statistiques du corpus, incluant le nombre d'AC et d'AAD ainsi que la longueur moyenne en nombre de mots des descriptions.

AC		AAD		
Nb	**Longueur moy. (nb mots)**	**Cours**	**Nb**	**Longueur moy. (nb mots)**
79	7.54	Algorithmique	76	7.76
		Java	40	4.68
		Base de Données	44	10.89
		Électronique	20	9.10
		Python	30	6.13
		Analyse Numérique	51	7.79
		Trait. du Signal	17	5.56
		Compilation	24	10.72

TABLEAU 3 – Statistiques descriptives du corpus

3.1 Méthodologie

Notre méthodologie repose sur quatre étapes principales, illustrées dans la figure 2.

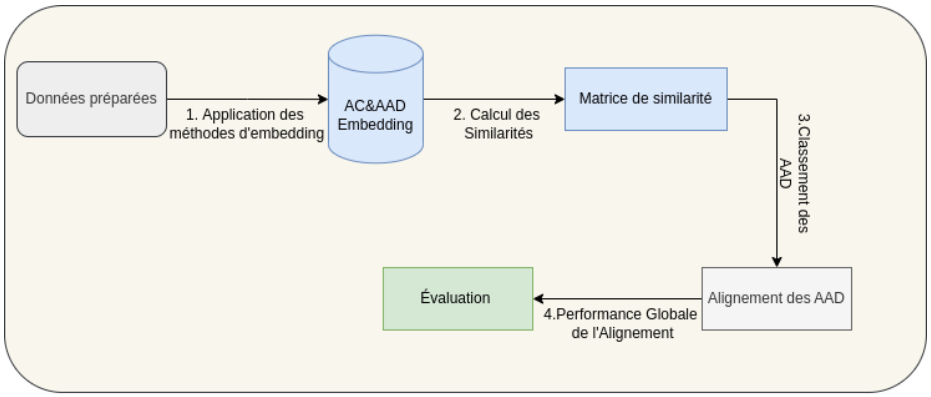

FIG. 2 – Diagramme des étapes de notre méthodologie adoptée.

Étape 1 : Application des méthodes d'Embedding. Quatre modèles issus de la section 2.1 ont été sélectionnés.

Pour les *embeddings* statiques, Word2Vec et FastText ont été appliqués après tokenisation. Word2Vec utilise l'algorithme Skip-Gram avec une dimension vectorielle de 300 et une fenêtre contextuelle de taille 5. FastText, configuré avec une dimension de 100 et une fenêtre de 5, intègre des *n-grammes* pour mieux gérer les variations morphologiques.

Les représentations de phrases ont été obtenues en moyennant les vecteurs des mots.

Pour les *embeddings* dynamiques, nous utilisons `SBERT` (préentraîné sur des données multilingues) et `sentence-camembert-large`, un modèle français optimisé pour la similarité de phrases. Ces modèles produisent des vecteurs contextuels de dimension 768.

Étape 2 : Calcul des similarités. Une matrice de similarité est construite, chaque ligne représentant un AC, chaque colonne un AAD. Les scores sont calculés à l'aide de la similarité cosinus et représentent la proximité sémantique entre les deux éléments. Un extrait de cette matrice est présenté dans le tableau 4.

Étape 3 : Classement des AAD. Pour chaque AC, les AAD sont triés selon leur score de similarité décroissant. Ce classement permet de déterminer les alignements potentiellement pertinents.

Étape 4 : Évaluation des performances. À chaque couple (AC, AAD), une vérité terrain (1 si aligné, 0 sinon) est associée. Ces annotations sont comparées au classement généré pour construire les courbes ROC et calculer les scores AUC. L'AUC moyen est ensuite utilisé pour évaluer globalement chaque modèle. Un histogramme des AUC individuels permet enfin d'analyser les performances selon les AC.

AC	**AAD1**	**AAD2**	**AAD3**	**AAD4**	**AAD5**
specifier-1.1 ("Décomposer et organiser un processus")	0.70	0.67	0.60	0.24	0.31
specifier-1.3 ("Utiliser un langage formel")	0.62	0.19	0.43	0.74	0.70

TABLEAU 4 – Extrait de matrice de similarité entre les AC "specifier-1.1" et "specifier-1.3" (voir tableau 1) et leurs AAD associés.

4 Résultats

Les performances des différentes méthodes d'*embedding* sont résumées dans le tableau 5, qui présente les scores AUC moyens obtenus pour huit cours.

On observe que les modèles dynamiques surpassent systématiquement les modèles statiques. En particulier, `SBERT`, utilisé ici dans sa version multilingue, obtient les meilleures performances dans les cours d'Algorithmique (0.79), Java (0.69) et Base de Données 0.63. Le modèle `sentence-camembert-large`, spécifiquement conçu pour le français, surpasse `SBERT` dans plusieurs cas : il atteint les meilleures performances en Électronique (0.82), en Analyse Numérique (0.62), en Traitement du Signal (0.61), en Compilation (0.72) et à égalité en Python (0.67).

Ces résultats confirment l'intérêt des *embeddings* dynamiques, et plus précisément des modèles contextualisés, pour capturer des nuances sémantiques fines dans les descriptions courtes et techniques des AC et des AAD. En comparaison,

Cours	Embeddings Statiques		Embeddings Dynamiques	
	Word2Vec	**FastText**	**sbert**	**Sentence-camembert-large**
Algorithmique	0.54	0.55	**0.79**	0.74
Java	0.49	0.56	**0.69**	0.65
Base de données	0.49	0.51	**0.63**	0.60
Électronique	0.48	0.54	0.73	**0.82**
Python	0.54	0.59	0.67	**0.67**
Analyse numérique	0.52	0.586	0.60	**0.62**
Traitement du signal	0.45	0.45	0.49	**0.61**
Compilation	0.56	0.55	0.63	**0.72**

TABLEAU 5 – Scores AUC moyens obtenus pour les embeddings statiques et dynamiques dans huit cours.

les modèles statiques comme Word2Vec et FastText produisent des résultats nettement plus faibles, avec des scores AUC généralement inférieurs à 0.60.

La figure 3 présente une comparaison des scores AUC pour chaque AC du cours d'Électronique, obtenus avec les modèles `SBERT` et `sentence-camembert-large`. Les résultats montrent une supériorité systématique de `sentence-camembert-large`, notamment pour *specifier-1.1* $(0,91$ contre $0,67)$, *modeliser-1.2* $(0,96$ contre $0,87)$ ou encore *developper-1.1* $(0,90$ contre $0.75)$.

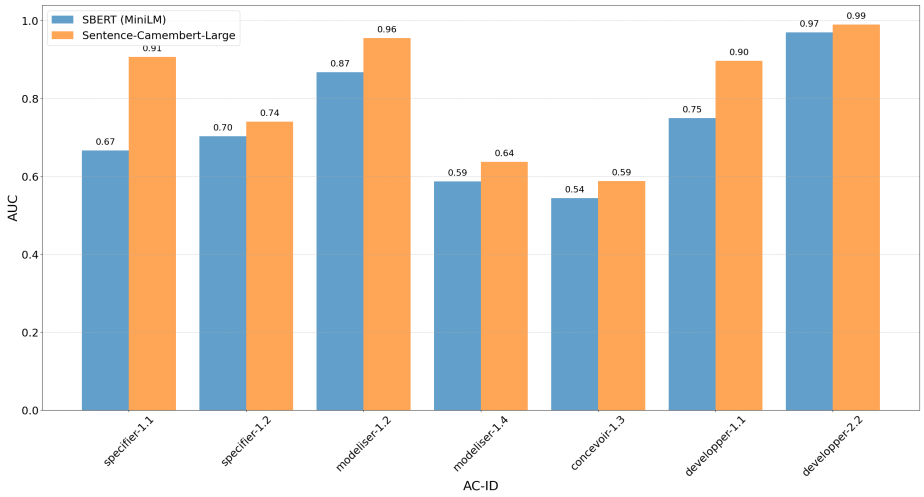

FIG. 3 – Comparaison des scores AUC pour chaque AC du cours Électronique avec `SBERT` et `sentence-camembert-large`.

Ces variations peuvent être attribuées à plusieurs facteurs. Le premier facteur concerne le nombre d'AAD associés à un AC par les responsables de cours. Par exemple, l'AC *developper-2.2* n'est associé qu'à deux AAD par l'enseignant. Le second facteur, qui est une hypothèse, concerne l'aspect explicite des AC et des AAD : le nombre de mots, l'implicite, la polysémie, etc.

5 Conclusion

Nous avons proposé une méthode d'alignement sémantique entre les AC et les AAD, dans le cadre de l'Approche Par Compétences. Le problème est formulé comme une tâche de recherche d'information, fondée sur des *embeddings* vectoriels.

L'évaluation menée sur huit cours montre la supériorité des modèles d'*embedding* dynamiques, notamment `SBERT` et `sentence-camembert-large`. Ces *embeddings* semblent être capables de capturer des relations contextuelles complexes, malgré des descriptions courtes et un déséquilibre important entre AC et AAD.

Comme perspectives, nous prévoyons de reformuler certains AC et AAD pour être plus explicite et donc améliorer la qualité des alignements. Cette hypothèse fait l'objet de premiers travaux encourageants.

Références

1. Basque, J., Rogozan, D. : Vers une méthode d'ingénierie de l'approche-programme en enseignement supérieur (2013)
2. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T. : Enriching word vectors with subword information. In : Transactions of the Association for Computational Linguistics. vol. 5, pp. 135–146 (2017)
3. Cer, D., Yang, Y., Kong, S.y., Hua, N., Limtiaco, N., St. John, R., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., et al. : Universal sentence encoder. In : Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing : System Demonstrations. pp. 169–174 (2018)
4. Chauvigné, C., Coulet, F. : L'approche par compétences dans l'enseignement supérieur : quelles fonctions pour l'enseignant ? (2010)
5. Chevallet, J.P. : Recherche d'information. Eyrolles (2013)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K. : Bert : Pre-training of deep bidirectional transformers for language understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies pp. 4171–4186 (2019)
7. Şeref Kerem ÇORBACIOĞLU, AKSEL, G. : Receiver operating characteristic curve analysis in diagnostic accuracy studies : A guide to interpreting the area under the curve value. Turkish journal of emergency medicine (Online) **23**(4) (2023)
8. Lahitani, S., et al. : Cosine similarity as a measure of textual proximity in information retrieval systems. International Journal of Information Retrieval Research (2016)

9. Manning, C.D., Raghavan, P., Schütze, H. : Introduction to Information Retrieval. Cambridge University Press (2008)
10. Mikolov, T., Chen, K., Corrado, G., Dean, J. : Efficient estimation of word representations in vector space (2013)
11. Pennington, J., Socher, R., Manning, C.D. : Glove : Global vectors for word representation. Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) pp. 1532–1543 (2014)
12. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L. : Deep contextualized word representations. arXiv preprint arXiv :1802.05365 (2018)
13. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, T., Matena, M., Zhou, Y., Li, W., Liu, P.J. : Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research **21**(140), 1–67 (2020)
14. Reimers, N., Gurevych, I. : Sentence-bert : Sentence embeddings using siamese bert-networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing pp. 3982–3992 (2019)
15. Salle, A., Villavicencio, A., Idiart, M. : Matrix factorization with additional constraints for improved word representations. In : Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics : Volume 2, Short Papers. pp. 230–236 (2016)
16. Tardif, J. : L'évaluation des compétences : Documenter le parcours de développement (2006)
17. Wang, S., Zhou, W., Jiang, C. : A survey of text representation and embedding techniques in nlp. IEEE Access **8**, 69617–69637 (2020)