Actes de la 28e Conférence sur le Traitement Automatique des Langues Naturelles
Lille, France, 28 juin au 2 juillet 2021

Mesure de similarité textuelle pour l'évaluation automatique de copies d'étudiants

Xiaoou Wang [1] Xingyu Liu [1] Yimei Yue [2]
(1) Université Paris Nanterre, France
(2) Inalco, France
xiaoouwangfrance@gmail.com

RESUME

Cet article décrit la participation de l'équipe Nantalco à la tâche 2 du Défi Fouille de Textes 2021 (DEFT) : évaluation automatique de copies d'après une référence existante. Nous avons utilisé principalement des traits basés sur la similarité cosinus des deux vecteurs représentant la similarité textuelle entre des réponses d'étudiant et la référence. Plusieurs types de vecteurs ont été utilisés (vecteur d'occurrences de mots, vecteur tf-idf, embeddings non contextualisés de fastText, embeddings contextualisés de CamemBERT et enfin Sentence Embeddings Multilingues ajustés sur des corpus multilingues). La meilleure performance du concours sur cette tâche a été de 0.682 (précision) et celle de notre équipe 0.639. Cette performance a été obtenue avec les Sentence Embeddings Multilingues alors que celle des embeddings non ajustés ne s'est élevée qu'à 0.55, suggérant que de récents modèles de langues pré-entraînés doivent être fine-tunés afin d'avoir des embeddings adéquats au niveau phrastique.

ABSTRACT

Textual similarity measurement for automatic evaluation of students' answers
This paper describes the Nantalco team's participation in task 2 of the DEFT contest in 2021: Automatic evaluation of students' answers based on an existing reference. We mainly used features based on the cosine similarity of the two vectors representing the textual similarity between student responses and the reference. Several types of vectors were used (count vector, tf-idf vector, non-contextualized embeddings from fastText, contextualized embeddings from Camembert and fine-tuned Multilingual Sentence Embeddings). The best performance of the contest on this task was 0.682 (precision). The maximum performance of our team (0.639) was obtained with the Multilingual Sentence Embeddings, while the best performance of raw embeddings of CamemBERT was only 0.55, suggesting that recent pre-trained language models need to be fine-tuned in order to have adequate embeddings at the sentence level.

MOTS-CLES : évaluation automatique, similarité textuelle, CamemBERT
KEYWORDS: automatic evaluation, textual similarity, CamemBERT

Atelier DÉfi Fouille de Textes (DEFT), pages 63–71.

1. Description de l'équipe et de la distribution des tâches

L'équipe Nantalco est composée de trois Talistes issus de deux établissements : Université Paris Nanterre et Inalco. Parmi les trois tâches proposées (Grouin et al., 2021), nous avons choisi la deuxième tâche visant à évaluer automatiquement des copies d'étudiants d'après une réponse référence fournie par l'enseignant. Le responsable Xiaoou Wang s'est chargé du test des features et de la rédaction de l'article final, Xingyu Liu de l'exploration statistique du corpus et Yimei Yue de la normalisation des données.

2. Quelques caractéristiques du corpus d'entraînement

2.1. Nature de la référence

Les références données par l'enseignant sont de deux types : réponses référence et barèmes. Le premier type de réponses permet directement une mesure de similarité textuelle, le deuxième type requiert un traitement manuel ou un système de traduction qui convertit les barèmes en réponses. Comme le nombre de références du type barèmes est peu élevé, nous avons procédé à une conversion manuelle. A titre d'exemple, trois réponses ont été créées manuellement à partir de la question 1004 où les barèmes sont « head 1, html 0 et meta 0.5 ». Ces réponses créées manuellement ont été intégrées dans le corpus des réponses d'étudiants avec le numéro d'étudiant 0.

« Question 1004 Quelle balise HTML contient les informations destinées au navigateur et aux moteurs de recherche ? head 0 si html 0.5 si meta »

Cette phase manuelle nous a permis d'augmenter le nombre de réponses de 3820 à 3861.

2.2. Répartition des classes dans le jeu de données

Il est important d'examiner la distribution des classes avant de tester des modèles de classification. La FIGURE *1* montre une distribution bimodale avec un nombre important de notes 0 et 1. Nous sommes dès lors confronté à quelle hypothèse il faut faire concernant la distribution des notes dans le corpus test. Dans le cas d'un échantillonnage biaisé, il est possible de faire du downsampling qui consiste à ne tenir compte que d'un sous-ensemble de la classe dominante. Dans le cas des classifieurs comme SVM, il est aussi possible d'ajuster le paramètre C pénalisant davantage une mauvaise classification en faveur de la classe dominante. Cependant, nous avons fait le choix de ne pas tenir compte de ce déséquilibre car il s'agit ici d'une habitude de notation susceptible de se reproduire dans les données de test.

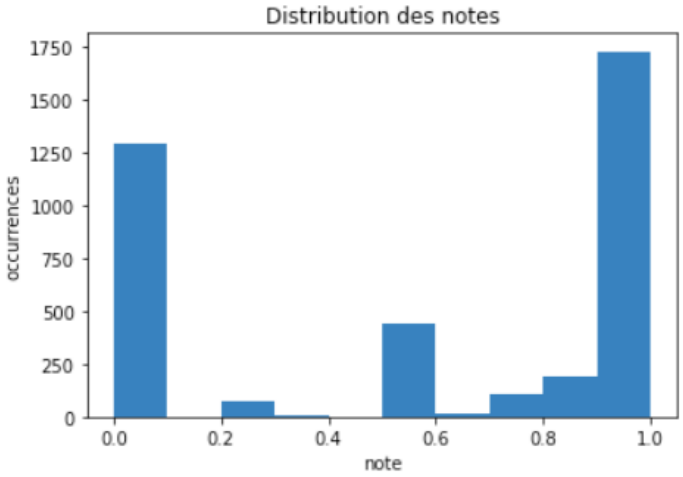

FIGURE 1 : Distribution des notes dans le corpus d'entraînement

3. Normalisation des données

Les processus de normalisations que nous avons effectués sur le corpus sont :

- suppression des tags html entourant les corrections et réponses
- suppression des mots vides avec des listes de mots-vides proposées par 3 bibliothèques : *NLTK* (Bird, 2006), *Spacy* (Honnibal et al., 2020) et *stopwordsiso*[1]
- suppression des mots vides basées sur des POS tags avec la bibliothèque *Stanza* (Qi et al., 2020), seuls les noms, nombres, noms propres, verbes, adverbes et adjectifs ont été gardés
- lemmatisation des tokens restants
- remplacement des espaces multiples avec un seul espace

Nous avons fait en sorte que le mot « pas » soit gardé dans tous les textes car la compréhension de la négation est importante pour la tâche de notation.

Notons que cette normalisation est également révélatrice de notre hypothèse qui suppose que la notation se fait en fonction du nombre de tokens apparaissant à la fois dans la réponse d'étudiant et dans la référence. Cette approche est approximative car dans ce cas les synonymes seront considérés comme des mots non pertinents alors que le fait d'utiliser un mot de sens proche est peu susceptible d'avoir un impact significatif sur la note finale. Des features basés sur des word embeddings seront par la suite ajoutés car ces derniers sont plus robustes dans le cas de l'usage des synonymes.

[1] https://pypi.org/project/stopwordsiso/

4. Élaboration des features

Les features que nous avons élaborés ont tous pour objectif de mesurer la similarité sémantique entre deux textes. Nous distinguons deux grandes classes.

4.1. Features basés sur la statistique textuelle

En utilisant le corpus normalisé, nous avons élaboré quelques features de base en utilisant de simples statistiques textuelles dont les définitions sont les suivantes :

1. ratio d'overlap : le nombre de mots communs apparaissant dans la réponse et la correction divisé par le nombre total de mots dans la correction
2. similarité cosinus calculée sur les vecteurs de compte de la réponse et de la correction
3. similarité cosinus calculée sur les vecteurs tf-idf de la réponse et de la correction
4. différence de nombre de caractères
5. différence de nombre de mots

4.2. Features basés sur des embeddings

Contrairement au comptage de mots qui est basé sur une représentation discrète (une occurrence de voiture n'est pas comparable avec une occurrence de bolide), les word embeddings (Mikolov et al., 2013) constituent une représentation continue de n dimensions qui permettent de mesurer la similarité lexicale. Pour représenter un document, il existe des méthodes telle que doc2vec (Le & Mikolov, 2014) qui, en dehors des vecteurs lexicaux, génère aussi un vecteur pour chaque document. Mais vu la longueur des réponses/références du corpus et la quantité de données d'entraînement, nous avons opté pour la méthode la plus simple consistant à faire la moyenne de tous les vecteurs.

Il existe principalement deux types d'embeddings lexicaux. Des embeddings non contextualisés proposent un seul vecteur pour un mot donné et des embeddings qui varient en fonction du contexte. Nous avons respectivement utilisé fastText (Bojanowski et al., 2017) et CamemBERT (Martin et al., 2019) pour générer ces embeddings en français. Pour ce dernier, il est aussi courant d'utiliser le token spécial [cls] pour représenter un document.

Avant de procéder à l'entraînement proprement dit, nous voudrions, à l'aide d'un simple exemple, montrer la non pertinence de la moyenne des embeddings CamemBERT pour représenter le sens des phrases. Nous utilisons à cet effet les 3 phrases suivantes :

1. J'aime les chats.
2. Je déteste les chats.
3. J'adore les chats.

Deux embeddings ont été utilisées pour calculer la similarité textuelle de ces phrases grâce à la similarité cosinus : la moyenne des embeddings et l'embedding du token [cls]. Notez que les fonctions permettant ces comparaisons ont été publiées dans le package *frenchnlp*[2], développé par notre équipe afin de faciliter l'utilisation des modèles de langue du type CamemBERT. La TABLE *1* montre les résultats :

Paire de phrases	Similarité cosinus basée sur la moyenne des embeddings	Similarité cosinus basée sur le token [cls]
j'aime vs je déteste	0.91	0.99
j'aime vs j'adore	0.99	1

TABLE 1 : Score de similarité entre 3 phrases en utilisant la moyenne des embeddings et [cls]

Il est évident que la différence de polarité entre phrase 1 et phrase 2 est mal représentée par cette méthode. Ce problème a été décrit plus en détail dans (Reimers & Gurevych, 2019), où les embeddings de BERT ont été comparés avec les embeddings Glove (Pennington et al., 2014) sur le benchmark STS (Cer et al., 2017), un corpus de phrases multilingue annoté en similarité textuelle. La mesure de la performance est la corrélation de Spearman et la performance de la moyenne des embeddings de BERT est seulement 46.35 contre 58.02 des embeddings GloVe. Les auteurs ont ensuite affiné les embeddings sur plusieurs benchmarks (dont STS), ce qui aboutit à une performance montée jusqu'à 88.77. Le package *sentence-transformers*[3], créé par les auteurs, permet de générer des sentence embeddings basés sur la méthode décrite dans l'article.

Il n'existe pas de sentence embeddings en français entraînés sur des corpus unilingues, cependant un modèle multilingue entraîné sur un corpus de plus de 50 langues, stsb-xlm-r-multilingual, est disponible. La TABLE *2* montre les résultats de ce modèle.

Paire de phrases	Similarité cosinus basée sur les sentence embeddings multilingues
j'aime vs je déteste	0.46
j'aime vs j'adore	0.96

TABLE 2 : Score de similarité entre 3 phrases en utilisant des sentence embeddings multilingues

[2] https://pypi.org/project/frenchnlp/, state of the art toolkit for Natural Language Processing in French.

[3] https://www.sbert.net/

Grâce à ces nouveaux embeddings, la différence de polarité est mieux représentée.

Par manque de corpus pour tester l'efficacité de ce modèle multilingue à plus grande échelle, nous avons décidé de tester toutes les 4 méthodes de représentation phrastique sur notre tâche.

4 sets de features au total ont été donc établis :

1. Les 5 features de base + similarité cosinus basée sur la moyenne des embeddings fastText
2. Les 5 features + similarité cosinus basée sur la moyenne des embeddings CamemBERT
3. Les 5 features + similarité cosinus basée sur l'embedding du token [cls]
4. Les 5 features de base + similarité cosinus basée sur les sentence embeddings de stsb-xlm-r-multilingual

5. Entraînement

Nous avons considéré la tâche comme une tâche de classification. Notons qu'il est tout à fait possible de modéliser cette tâche avec la régression linéaire, mais la dominance des notes 0 et 1 dans le jeu de données et la nature quasi-discrète des notes (il n'y pas de notes du type 0.11 ou 0.235) nous ont conduits à considérer les notes comme des chaînes de caractères et non des valeurs numériques continues.

Nous avons décidé d'utiliser SVM car ce dernier peut être ajusté avec des fonctions de classification différentes (astuce du noyau) permettant de capturer des relations non linéaires entre features et classes. Outre le noyau linéaire, nous avons aussi testé le noyau polynomial (degré 3) et le noyau RBF.

Pour la séparation des données train et dev, nous avons procédé à 10 folds stratifiés, générant ainsi 10 splits où le pourcentage de chaque classe est respecté. La moyenne de la précision sur les 10 splits a été utilisée comme indice de la performance du modèle.

6. Résultats et discussions

Toutes choses égales par ailleurs, la performance du noyau RBF a été systématiquement meilleure que les deux autres. Nous avons de ce fait présenté, dans la TABLE *3*, uniquement les résultats issus de ce kernel.

	fastText	Moyenne CamemBERT	Embedding [cls] CamemBERT	stsb-xlm-r-multilingual
Précision sur 10 splits	0.53	0.55	0.53	0.67

TABLE 3 : Précision en moyenne calculée d'un SVM au noyau RBF sur 10 splits du corpus.

Les résultats montrent que le score de similarité calculé avec des sentences embeddings ajustés sur un corpus multilingue (stsb-xlm-r-multilingual), conjointement avec les 5 features de base mentionnés plus haut, permettent de mieux prédire les notes des étudiants, alors qu'il n'y a pas d'écart significatif entre l'utilisation de la moyenne des embeddings fastText, des embeddings CamemBERT et du token [cls] de la phrase. Ces résultats vont dans le sens de l'article de (Reimers & Gurevych, 2019) qui montre que Bert, sans fine-tuning, n'est pas plus adéquat que les embeddings non contextualisés pour calculer la similarité sémantique entre deux phrases. En outre, nos résultats suggèrent que les sentence embeddings, bien que entraînés sur des corpus non unilingues, permettent d'obtenir des mesures de similarité plus pertinentes. Le modèle entraîné avec la similarité cosinus des embeddings de stsb-xlm-r-multilingual nous a permis d'atteindre une précision de 0.639 (meilleure performance de notre équipe) sur le jeu de données test.

Par contraintes de temps, nous n'avons pas pu tester d'autres types de sentence embeddings (cf. par exemple InferSent (Conneau et al., 2017) ou Universal Sentence Encoder (Cer et al., 2018)). Cependant, nous espérons avoir au moins montré la non pertinence des embeddings de CamemBERT à l'état brut pour représenter les phrases.

Un reproche pouvant être exprimé à l'égard de notre article est l'abandon des informations contenues dans les questions. En effet, il est possible de construire un système question-réponse où la référence d'enseignant peut être considérée comme un contexte et ensuite de calculer la probabilité d'une réponse spécifique. Des corpus en français comme FQuAD (d'Hoffschmidt et al., 2020) sont disponibles aujourd'hui et peuvent être utilisés afin de concevoir un tel système. Nous explorerons cette piste si l'occasion se présente.

Références

Bird, S. (2006). NLTK: The natural language toolkit. *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, 69–72.

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

Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., & Specia, L. (2017). Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *ArXiv Preprint ArXiv:1708.00055*.

Cer, D., Yang, Y., Kong, S., Hua, N., Limtiaco, N., John, R. S., Constant, N., Guajardo-Céspedes, M., Yuan, S., & Tar, C. (2018). Universal sentence encoder. *ArXiv Preprint ArXiv:1803.11175*.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., & Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. *ArXiv Preprint ArXiv:1705.02364*.

d'Hoffschmidt, M., Belblidia, W., Brendlé, T., Heinrich, Q., & Vidal, M. (2020). FQuAD: French question answering dataset. *ArXiv Preprint ArXiv:2002.06071*.

Grouin, C., Grabar, N., & Illouz, G. (2021). Classification de cas cliniques et évaluation automatique de réponses d'étudiants: Présentation de la campagne DEFT 2021. *Actes de DEFT. Lille*.

Honnibal, M., Montani, I., Van Landeghem, S., & Boyd, A. (2020). *spaCy: Industrial-strength natural language processing in python*. Zenodo. https://doi.org/10.5281/zenodo.1212303

Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. *International Conference on Machine Learning*, 1188–1196.

Martin, L., Muller, B., Suárez, P. J. O., Dupont, Y., Romary, L., de la Clergerie, É. V., Seddah, D., & Sagot, B. (2019). Camembert: A tasty french language model. *ArXiv Preprint ArXiv:1911.03894*.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *ArXiv Preprint ArXiv:1301.3781*.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543.

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., & Manning, C. D. (2020). Stanza: A Python natural language processing toolkit for many human languages. *ArXiv Preprint ArXiv:2003.07082*.

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *ArXiv:198.10084 [Cs]*. http://arxiv.org/abs/1908.10084