DEFT 2021: Évaluation automatique de réponses courtes, une approche basée sur la sélection de traits lexicaux et augmentation de données

Timothée Poulain, Victor Connes

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

DEFT 2021: Évaluation automatique de réponses courtes, une approche basée sur la sélection de traits lexicaux et augmentation de données

Timothée Poulain[1] Victor Connes[1]
(1) Université de Nantes, LS2N, 2 Chemin de la Houssinière, 44300 Nantes, France
timothee.poulain@univ-nantes.fr, victor.connes@univ-nantes.fr

RÉSUMÉ

Cet article présente la participation de l'équipe Proofreaders du LS2N au DÉfi Fouille de Textes 2021 (DEFT 2021). La tâche proposée consiste en la poursuite automatique de l'évaluation de réponses courtes d'étudiants (EAQRC) à partir de quelques réponses déjà corrigées par l'enseignant pour chaque énoncé. Une étude comparative de différents traits lexicaux, ainsi qu'une augmentation artificielle de données et de différents modèles de régression pour la notation des réponses courtes est réalisée. Les méthodes sont évaluées en termes de précision, d'erreur quadratique moyenne et de score de corrélation de Spearman. Notre erreur quadratique moyenne varie entre 0.090 et 0.101 et notre précision entre 0.147 et 0.17.
Le code source est disponible à l'adresse suivante : https://github.com/poulain-tim/DEFT_2021

ABSTRACT

DEFT 2021 : Automatic short answer grading, a lexical features selection and data augmentation based approach.

This paper presents the participation of the LS2N Proofreaders team at DÉfi Fouille de Textes 2021 (DEFT 2021). The proposed task consists of Automatic Short Answer Grading (ASAG) continuation from a few answers already corrected by the teacher for each question. A comparative study of different lexical features, as well as artificial data augmentation and different regression models for grading short answers, has been performed. The methods are evaluated in terms of accuracy, mean square error, and Spearman correlation score. Our mean squared error ranges from 0.090 to 0.101 and the accuracy between 0.147 and 0.17.
The source code is available at the following address, https://github.com/poulain-tim/DEFT_2021

MOTS-CLÉS : Questions à réponses courtes (QRC), Evaluation automatique des réponses courtes, e-learning, apprenant, DEFT, sélection de caractéristiques.

KEYWORDS: Short answer questions (SAQ), Automatic short answer grading (ASAG), e-learning, learner, DEFT, feature selection.

1 Introduction

Le développement et l'augmentation du nombre de cours dans des plate-formes et formations digitales dans la sphère éducative traduisent un renouvellement des modalités d'apprentissage (Acquatella, 2018). Le modèle d'apprentissage évoluant, il faut adapter un modèle d'évaluation en conséquence.

Atelier DÉfi Fouille de Textes (DEFT), pages 31–40.

L'assimilation de compétences se matérialise par des tests sur les connaissances de l'apprenant. Dans cette pluralité de stratégies d'évaluation, questionnaire à choix multiples, question à choix uniques, nous nous concentrons sur les questions à réponses courtes. L'épreuve de QRC (questions à réponses courtes) est reconnue comme un dispositif d'évaluation performant permettant à l'élève de formuler ces propres réponses, mais aussi au professeur d'évaluer l'argumentaire, la rédaction et la synthétisation. Contrairement aux questionnaires à choix multiples pour lesquels la restitution automatique des résultats se fait de manière précise et aisée, l'évaluation automatique des questions à réponses courtes est un domaine de recherche à part entière, comme en témoigne l'étude comparative de Galhardi & Brancher (2018) qui prend en compte 44 systèmes différents d'évaluation des QRC. C'est dans ce cadre, que la campagne d'évaluation scientifique francophone, DÉfi et Fouille de textes (DEFT 2021) (Grouin *et al.*, 2021) proposent deux tâches sur la correction automatique de copies électroniques d'étudiants. La première est une évaluation automatique des réponses d'après une référence professorale existante (Tâche n°2) et la deuxième est une poursuite automatique de l'évaluation de réponses d'étudiants à partir de premières évaluations suivant des questions aussi bien ouvertes que fermées (Tâche n°3). Dans cet article, seulement l'évaluation de la tâche n°3 est effectuée.

2 Les méthodes d'évaluation automatique des questions à réponses courtes

Le domaine de l'évaluation automatique des questions à réponses courtes (EAQRC) concerne toutes les approches essayant d'attribuer automatiquement une note à une tentative de réponse courte à une question courte. Les réponses et les questions ne doivent pas dépasser un paragraphe. Habituellement, les corpus sont donc composés de questions posées par des enseignants et de plusieurs tentatives de réponses évaluées par une note. Certaines autres informations peuvent y figurer telles qu'un barème, des réponses de référence ou des éléments de correction. Les ensembles de données habituels pour cette tâche sont [1] [2] [3] [4].

L'approche habituelle pour les EAQRC consiste à extraire des caractéristiques à la fois des réponses et des questions en utilisant des techniques de traitement automatique du langage naturel (TALN) et du web sémantique (WS). La note est ensuite prédite à partir des traits extraits par un algorithme d'apprentissage automatique. Les auteurs de l'étude comparative de ces méthodes (Galhardi & Brancher, 2018) distinguent trois grands types de traits : Lexicaux, Syntaxiques, Sémantiques. Nous faisons un tour d'horizon de ces différents traits à partir des articles suivants Galhardi & Brancher (2018), Galhardi *et al.* (2018), Sultan *et al.* (2016).

Parmi les traits lexicaux, les modèles les plus couramment utilisés sont les n-grammes (y compris les Sac de mots), les méthodes de pondération (TF-IDF) et les méthodes de plongement de mots (principalement Word2Vec, FastText). Nous distinguons dans cet ensemble de traits, des statistiques intrinsèques liées à la réponse (R) et des statistiques exploitant la similarité au niveau lexical entre la réponse traitée (R), sa question associée (Q) et les réponses de référence (RR). Pour les traits des

1. http://web.eecs.umich.edu/~mihalcea/downloads.html
2. www.uni-tuebingen.de/en/research/core-research/collaborative-research-centers/sfb-833/section-a-context/a4-meurers/software-resources-and-corpora.html
3. www.kaggle.com/c/asap-sas
4. www.cs.york.ac.uk/semeval-2013/task7/index.php%3Fid=data.html

statistiques textuelles intrinsèques, on retrouve fréquemment, le nombre de mots, de mots uniques, de caractères, de verbes, de mots mal orthographiés, de longueur moyenne des mots et des phrases, ... À partir de ces statistiques, on infère un certain nombre de rapports tel que la longueur moyenne des mots, le ratio nombre de mots uniques par nombre de mots. On peut aussi s'appuyer sur les statistiques des réponses de références ou de la question pour induire le rapport de longueur (en nombre de mots ou de caractères) entre R/RR ou entre R/Q. Un autre groupe de trait est celui qui utilise une métrique pour mesurer la similarité lexicale entre R/Q et R/RR. Ce groupe se décompose souvent en plusieurs sous-groupes, on y retrouve les mesures de similarité sur les sacs de mots comme les distances Cosine, Euclidienne, Recouvrement, Sorensen-Dice, les similarités établies par les distances d'éditions, Levenshtein, Hamming, Jaro-winkler. On y adjoint les mesures de similarité au niveau de la séquence, tel que LCS (en anglais *longest common subsequence*) ou Ratcliff-Obershelp et les similarités à partir d'algorithme de compression sans perte comme la transformée de Burrows-Wheeler, très utile pour la détection de répétition.

Les traits syntaxiques étudiés sont les étiquetages morphosyntaxiques et leurs similarités dérivées. Les n-grammes, communément utilisés pour modéliser le modèle de langage pour la tâche de EAQRC (Burrows *et al.*, 2015) sont généralement extraits avec un parseur tel que Stanford Parser [5]. On peut aussi extraire un triplet contenant deux mots et leur relation de dépendance.

Enfin, les traits sémantiques sont les caractéristiques inhérentes à un concept, au sens du mot, ayant vocation à déterminer le type de relations lexicales qui existe entre les mots dans une langue et facilitant l'accès sémantique aux mots. Les mesures de similarité à partir de connaissances externes peuvent être considérées comme telles. Ce sont par exemple les méthodes d'étiquetage des rôles sémantiques où l'on attribue des étiquettes aux rôles que les mots représentent dans les phrases en tenant compte de leur aspect sémantique ou encore l'utilisation de la base de données lexicales "Wordnet" pour enrichir et mettre en relation le contenu lexical et sémantique de la langue. D'autres mesures de similarité existent, mais sont cette fois fondées sur le corpus, (Gabrilovich *et al.*, 2007) a démontré l'intérêt d'utiliser une mesure basée sur un corpus (LSA) entraînée sur un corpus spécifique à un domaine. D'autres méthodes acquièrent des informations statistiques pour calculer la relation entre les mots et les documents comme ESA (en anglais *Explicit Semantic Analysis*) et DISCO (en anglais *Extracting DIStributionally similar words using COoccurrences*). Un dernier type de traits sémantiques existe, l'implication textuelle (en anglais *textual entailment*), une méthode qui consiste à juger si un texte peut être déduit d'un autre texte.

D'autres approches plus récentes d'apprentissage profond qui ne sont pas traitées dans cet article ont vu le jour tel que les auto-encoder (Yang *et al.*, 2018), les architectures siamoises (Kumar *et al.*, 2017) ou les modèles à base de transformers pré-entrainés (BERT) par ajustement (*finned tunned*) sur la problématique de EAQRC (Sung *et al.*, 2019).

3 Données

La tâche demandée dans le cadre de cette édition du défi est une tâche de poursuite automatique de l'évaluation de réponses d'étudiants à partir de premières évaluations. Les corpus utilisés se composent d'une centaine d'énoncés en informatique spécialisés en programmation web et bases de données ainsi que des réponses produites par une cinquantaine d'étudiants en moyenne par question, sur deux années d'enseignement. Deux types de question/réponse apparaissent dans ces ensembles de

5. https://nlp.stanford.edu/software/lex-parser.shtml.

données (voir Table 1), les fermées où une réponse syntaxiquement très précise est attendue et les ouvertes où une réponse sémantiquement précise est plus préconisée. Le domaine spécifique de la programmation influence certain de nos choix et nos hypothèses de pré-traitements des données.

Type de question	Exemple de question	Exemple de réponse
Question ouverte	Accessibilité : Indiquez trois précautions à prendre pour garantir [..]	L'enregistrer dans un format qui est couramment utilisé (exemple : .txt) afin de [..]
Question fermée	Donnez le code HTML complet créant le formulaire contenant les champs [...]	<html> <head> <title> Les UEs </title> </head> <body> [...] </body> </html>

TABLE 1 – Exemples de question/réponse pour chaque type de questions

Les données mises à disposition par les organisateurs (Grouin *et al.*, 2021) du défi DEFT 2021 sont composées de deux corpus. Les deux corpus sont séparés au préalable en base d'apprentissage et de test selon les proportions 2/3 et 1/3. L'ensemble d'apprentissage de la tâche n°2 est composé de 50 questions et 50 éléments de réponses du professeur. Parmi ces 50 questions, 30 ont 116 réponses d'étudiants, et les 20 questions restantes ont 17 réponses d'étudiants. L'ensemble de test de la tâche n°2 suit la même logique et est présenté en Table 2.
Le second corpus correspond à la tâche n°3 est composé de 21 questions, sans éléments de réponses du professeur et en moyenne 69 réponses par question (avec un minimum de 41 et un maximum de 116). L'ensemble de test de la tâche n°3 suit la même logique avec en moyenne 9% de réponses d'étudiants déjà corrigées par l'enseignant et est présenté en Table 2.

Corpus	**Entrainement**		**Test**		
	Questions	*Réponses*	*Questions*	*Réponses*	*Réponses Corrigées*
Tâche n°2	50	3820	21	1644	–
Tâche n°3	11	769	6	387	0.09

TABLE 2 – Description des corpus d'entraînement et de test pour les deux tâches

4 Notre Approche

De par la spécificité et la taille de notre corpus, nous choisissons de nous focaliser sur les méthodes d'extraction de traits. En particulier, nous nous intéressons aux traits lexicaux comme présentés dans la section 2. Pour contre-balancer le manque de données, nous employons une méthode de génération artificielle de données, ainsi qu'une méthode de sélection de traits. Enfin en suivant les approches de la littérature scientifique, nous prédisons la note par un algorithme d'apprentissage automatique sur une tâche de régression. Dans les prochaines sections, nous détaillons les différentes modalités de notre approche.

4.1 Pré-traitements des données

Pour tenir compte des particularités linguistiques de notre corpus, nous appliquons un certain nombre de méthodes de pré-traitements aussi bien pour les réponses, que pour les questions. Nous remplaçons les chevrons ouverts et fermants de chaque balises HTML par des symboles "ó" et "ò" en vue de l'utilisation de la méthode de création de plongements lexicaux, FastText. Nous substituons l'ensemble des nombres par le mot NUM, dans le but de réduire notre vocabulaire sachant qu'aucune des questions ne requière de réponses numériques. La suppression de la casse, la suppression des

sauts de lignes "$\backslash n$" ont été aussi réalisées. Après quelques expérimentations, nous avons décidé de ne pas normaliser notre corpus en utilisant la racinisation. En effet, en disséquant les données, et en inspectant le thème informatique et spécialement en programmation web et base de données des énoncés, nous formulons plusieurs hypothèses. La première est la conservation de la ponctuation dans nos modèles. La syntaxe de la programmation web est précise et ponctuée par un ensemble de signes graphiques (deux points, point virgule, accolade, ...) . De surcroît, si nous supprimons celle-ci, cela peut poser problématique pour la notation automatique des questions fermées (par exemple, un type de balise) où chaque signe graphique a son importance. Pour les mêmes raisons, nous décidons de conserver les mots dans leurs flexions originelles et de ne pas faire de racinisation. Par exemple, si le mot "former" apparaît dans le texte, et que nous utilisons la racinisation il est transformé en "form", ce qui est problématique car sans ponctuation, ni racinisation, le mot "form" peut émaner d'une balise html.

4.2 Augmentation des données

Il y un adage en apprentissage automatique qui dit "Plus nous disposons de données, meilleures sont les performances que nous pouvons atteindre". Par conséquent, une augmentation appropriée des données est utile pour améliorer les performances de notre modèle. Néanmoins, acquérir et étiqueter des données supplémentaires manuellement est un travail long, coûteux et fastidieux. Très en vogue dans le domaine de vision par ordinateur, où l'on peut générer facilement d'autres images en la retournant, en la bruitant, la génération de données en TAL est plus compliqué en raison de la complexité du langage. En effet, chaque mot ne possède pas forcément un synonyme, et la substitution peut créer des ambiguïtés. Toutefois, pour notre cas spécifique, nous tentons d'augmenter artificiellement notre corpus de deux manières. La première manière est simplement d'ajouter à notre corpus d'entraînement les données de la tâche n°2, les réponses et les questions.

La deuxième manière a recours à une méthode d'augmentation de données fréquemment utilisée en TALN. Notre objectif ici est de créer pour chaque question et chaque réponse, des nouvelles analogues, les réponses générées héritant de la même note que l'original. Nous générons donc des questions et des réponses similaires à celles déjà présentes dans le jeu de données d'entraînement sauf que nous substituons à chaque entrée un ou plusieurs mots sémantiquement liés. Nous utilisons une méthode d'augmentation originalement proposée par Xie *et al.* (2020). L'idée de base est que les mots qui ont un score TF-IDF faible ne sont pas informatifs et peuvent donc être remplacés sans affecter la phrase. Pour ce faire, nous avons calculé l'ensemble des scores TF-IDF pour chaque mot sur le corpus d'entraînement (agrégation question et réponses) excluant "NUM" et "NO_ANS". Chaque question et chaque réponse constitue ici un document indépendant. Seuls les mots obtenant des scores TF-IDF moyens sur l'ensemble des documents inférieur à 1 (seuil manuellement fixé a partir de la distribution des scores TF-IDF) sont gardés comme candidat à la substitution. A l'aide de la méthode des K plus proche voisin (K-pvv) et d'un modèle de language FastText français, nous générons pour chaque mot un ensemble de mots candidats sur la base de leurs similarités en cosinus avec le mot originel dans l'espace des plongements et nous extrayons les 20 mots plus proches sémantiquement. Et enfin, un score pour chaque mot est calculé comme le produit entre la similarité cosinus entre le plongement du mot originel et le mot traité et un moins le TF-IDF du mot. (voir l'équation 1).

$$score(w) = \text{K-ppv}_{w_{origin},w}(embedding(w_{origin}, w)) \times (1 - \text{TF-IDF}(w)) \quad (1)$$

Cette méthode permet d'aboutir à un lexique de substitution n'affectant pas le sens avec leurs scores

associés, dont voici quelques exemples : ("septembre", "aout", 0.76), ("soucier", "préoccuper" , 0.74), ("continuellement", "constamment", 0.72), ("expliquer", "décriver", 0.70), ("certainement", "incontestablement", 0.63), ("provoque", "occasionne", 0.58), ("manières", "façons", 0.52), ("cependant", "paradoxalement", 0.50). [6]

Les nouvelles questions et réponses sont ensuite générées en remplaçant des mots des phrases originelles par les meilleurs candidats dans le lexique (de 1 à 2 mots).

En pratique nous avons choisi de remplacer au maximum un mot par question ou réponse et de créer 5 nouvelles questions ou réponses avec les 5 meilleurs mots candidats à la substitution.

Avec cette méthode nous obtenons au total 272 questions et 9153 réponses à partir des 67 questions et des 3736 réponses initialement présentes dans notre corpus d'entraînement.

4.3 Traits lexicaux

Ces traits sont aussi bien centrés sur les statistiques extraites de chaque réponse individuellement que les mesures de similarité pour R/Q et R/RR.

Traits liés au comptage : Nombre de mots, Nombre de chevrons ouverts et fermés, nombre de mots uniques, nombre de mots mal orthographiés (utilisant la bibliothèque python "SpellChecker"), le nombre de caractères, longueur moyenne des mots.

Ratios : Le rapport de longueur entre la réponse et les réponses de références exprimés en nombre de mots, le rapport de longueur entre la réponse et la question exprimés en nombre de mots.

Mesure de la similarité sur les sacs de mots : La distance Cosine, Recouvrement, Sorensen-Dice.

Mesure de la similarité sur la fréquence des mots : La distance Cosine.

Mesure de la similarité par distance d'édition : Distance de Levenshtein, Hamming, Jaro-Winckler.

Mesure de similarité au niveau de la séquence Taille de la plus longue séquence de caractère commune, similarité de Ratcliff Obershelp.

Si le nombre de réponses de références est supérieur à un, on extrait aussi bien le maximum, le minimum et la moyenne de la similarité entre R/RR pour chacune des mesures de similarité, distance d'édition etc.

Au total, nous avons extrait 60 traits différents.

4.4 Traits lexicaux issus des plongements de mots

Dans cette section, nous allons exploiter la représentation lexicale des mots dans un espace continu au lieu d'une représentation traditionnelle. Pour créer nos plongements de mots, nous avons une combinaison de notre corpus d'entraînement et d'un moissonnage de Wikilivres spécialisés [7]. Les Wikilivres sont une collection de textes pédagogiques libres rassemblés en livres et écrits en collaboration regroupant une large variétés de sujets.

6. Une liste exhaustive de ce lexique est disponible à https://github.com/poulain-tim/DEFT_2021/blob/main/DATA/corpus/substitutes.json

7. https://www.kaggle.com/dhruvildave/wikibooks-dataset

Pour notre application nous avons choisi ceux dont le titre contient un des mots clés suivants : "HTML", "JavaScript", "PowerPoint", "CSS". Nous avons restreint notre choix à des Wikilivres en français, pour créer notre corpus textuel nous agglomérons les informations du titre, du résumé et du corps de l'article aux questions et réponses de nos données. Ce corpus textuel est pré-traité (voir Section 4.1 est ensuite utilisé pour créer un modèle de language en utilisant la méthode FastText.

De plus, nous avons également testé une méthode de transfert d'apprentissage (transfert learning) en utilisant les vecteurs du modèle pré-entraînés FastText sur Common Crawl et Wikipedia français que nous spécialisons (fine tunnons) à l'aide de notre corpus textuel.

À partir de ces modèles de langage il est possible de calculer des distances entre les phrases. Soit deux phrases, $S_1 = w_1, w_2, ..., w_n$ et $S_2 = w'_1, w'_2, ...w'_m$. Pour mesurer la similarité lexicale entre S_1 et S_2, nous effectuons la moyenne des vecteurs de plongements de mots normalisés par la norme L2, en suivant l'implémentation de la bibliothèque c++ FastText [8]. Nous choisissons deux méthodes de similarité, la similarité euclidienne et la similarité cosinus. A l'instar des traits lexicaux sur la fréquence des mots, nous effectuons une similarité entre le plongement de la réponse traitée et le plongement de la question et entre le plongement de la réponse traitée et les N-plongements des réponses de référence. De la même façon, nous extrayons les scores de similarité minimale, moyenne et maximale par rapport aux N-réponses de référence que nous utilisons en tant que caractéristiques. La dernière caractéristique extraite est la moyenne de ces N-plongements des réponses de références par rapport au plongement de la réponse traitée.

Si nous ajoutons aux 60 traits lexicaux sur la fréquence des mots, les 9 traits lexicaux issus des plongements de mots, nous atteignons un total de 69 traits.

4.5 Sélection des traits

Après avoir créé de nombreux traits, nous choisissons le meilleur sous-ensemble de traits afin de réduire la dimensionnalité, contribuant à prédire de manière optimale notre note. Pour ce faire, nous utilisons l'algorithme Kbest de la bibliothèque python scikit-learn [9]. Cette méthode prend comme paramètre une fonction de score, dans notre cas *f_regression*, le F-score entre la note et le trait pour une tâche de régression. La fonction renvoie un tableau de score, un score par trait. SelectKBest sélectionne les k premières caractéristiques avec les scores les plus élevés. Nous faisons varier la valeur de k de 10 à 30.

Pour l'ensemble de nos expérimentations, nous utilisons la méthode des forêts aléatoires pour la tâche de régression de la librairie scikit-learn [10].

5 Résultats

À l'instar de Mohler *et al.* (2011), nous calculons un score de corrélation de Spearman et de l'erreur moyenne quadratique sur toutes les réponses des élèves de tous les ensembles de données. En effet,

8. https://fasttext.cc/
9. https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html
10. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

comme expliciter dans Burrows *et al.* (2015), nous pouvons voir le problème soit comme un problème de classification, soit comme un problème de régression. Chaque note maximale des questions n'étant pas normalisée en amont, nous pouvons nous retrouver avec une grande variété d'étiquettes de notes finales, dans notre cas, 21 différentes notes entre 0 et 1. En ce sens, nous choisissons d'aborder le problème comme une régression et non une classification, où l'utilisation de la mesure de l'erreur moyenne quadratique semble plus cohérente. Afin d'utiliser aussi la précision comme mesure d'évaluation, nous remplaçons le score prédit par la valeur de la note la plus proche observée dans les données d'entraînements.

La comparaison des résultats (voir tableau 3) montre l'avantage d'ajouter une à une les nouvelles méthodes au regard des trois métriques utilisées. L'amélioration en terme de précision va de 0.147 à 0.17. Toutefois, les différentes étapes impactent de manière variable les différentes métriques ce qui complexifie l'évaluation de l'apport de chacune de ces étapes. À titre d'exemple, SLP améliore grandement les résultats en termes de RMSE mais faiblement en terme de précision et de corrélation. De plus, nous nous sommes aperçus que les mesures de similarité engageant les questions n'étaient jamais retenues dans les traits pertinents sélectionnés.

Méthodes	RMSE	ρ	Précision
SLM	0.101	0.56	0.147
SLM + SLP	0.099	0.57	0.15
SLM + SLP+ AAD	0.090	0.62	0.152
SLM + SLP+ AAD + PP	0.093	0.61	0.17

TABLE 3 – Résultats sur le corpus de test de la tâche n°3 en fonction des différentes méthodes appliquées (SLM : Statistiques Lexicales sur le Mot, SLP : Statistique Lexicale pour les Plongements lexicaux, AAD : Augmentation Artificiel des Données, PP : Plongements lexicaux Pré-entraînés)

5.1 Résultats officiels

Nous présentons dans cette section les résultats officiels de la campagne DEFT 2021. Nous avons soumis trois exécutions pour la tâche n°3. Le tableau 3 résume nos trois exécutions. Nous constatons que la variante V1 obtient les meilleurs résultats. Les résultats sont légèrement modifiés par rapport au Tableau 3, compte tenu du changement des paramètres de racinisation, du nombre minimal de caractéristiques pour l'algorithme de KBest, du nombre maximal de réponses de références par réponse, ...

— Pour V1 : SLM + SLP + AAD + PP
— Pour V2 : SLM + SLP + AAD
— Pour V3 : SLM + SLP

Méthodes	RMSE	ρ	Précision
V1	0.093	0.61	0.17
V2	0.097	0.58	0.159
V3	0.133	0.60	0.133

TABLE 4 – Résultats de nos trois exécutions de modèles pour la tâche n°3

6 Conclusion

Dans ce travail, nous avons décrit la participation de l'équipe Proofreaders du LS2N à DEFT 2021. L'approche suivie par l'équipe modélise le problème EAQRC comme une tâche de régression, la solution proposée s'appuie sur l'extraction de traits lexicaux joint à une méthode de sélection de traits et une méthode d'augmentation de données textuelles. Les résultats obtenus montrent une corrélation entre les notes prédites et les notes de l'enseignant. Néanmoins, les résultats finaux restent faibles pour évaluer l'impact de chacune des étapes du processus. En perspective de ce travail, nous souhaiterions réévaluer l'apport de chacune des étapes de notre méthode de manière plus formelle.

Références

ACQUATELLA F. (2018). *Analyse stratégique du marché de la formation en ligne : les Moocs comme nouvelle variable des écosystèmes de plateformes digitales*. Thèse de doctorat, Paris, ENST.

BURROWS S., GUREVYCH I. & STEIN B. (2015). The eras and trends of automatic short answer grading. *International Journal of Artificial Intelligence in Education*, **25**(1), 60–117.

GABRILOVICH E., MARKOVITCH S. *et al.* (2007). Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJcAI*, volume 7, p. 1606–1611.

GALHARDI L. B. & BRANCHER J. D. (2018). Machine learning approach for automatic short answer grading : A systematic review. In *Ibero-american conference on artificial intelligence*, p. 380–391 : Springer.

GALHARDI L. B., DE MATTOS SENEFONTE H. C., DE SOUZA R. C. T. & BRANCHER J. D. (2018). Exploring distinct features for automatic short answer grading. In *Anais do XV Encontro Nacional de Inteligência Artificial e Computacional*, p. 1–12 : SBC.

GROUIN C., GRABAR N. & ILLOUZ G. (2021). Classification de cas cliniques et évaluation automatique de réponses d'étudiants : présentation de la campagne. In *Actes de DEFT. Lille.*

KUMAR S., CHAKRABARTI S. & ROY S. (2017). Earth mover's distance pooling over siamese lstms for automatic short answer grading. In *IJCAI*, p. 2046–2052.

MOHLER M., BUNESCU R. & MIHALCEA R. (2011). Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In *Proceedings of the 49th annual meeting of the association for computational linguistics : Human language technologies*, p. 752–762.

SULTAN M. A., SALAZAR C. & SUMNER T. (2016). Fast and Easy Short Answer Grading with High Accuracy. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, p. 1070–1075, San Diego, California : Association for Computational Linguistics. DOI : 10.18653/v1/N16-1123.

SUNG C., DHAMECHA T., SAHA S., MA T., REDDY V. & ARORA R. (2019). Pre-training bert on domain resources for short answer grading. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, p. 6073–6077.

XIE Q., DAI Z., HOVY E., LUONG M.-T. & LE Q. V. (2020). Unsupervised data augmentation for consistency training.

YANG X., HUANG Y., ZHUANG F., ZHANG L. & YU S. (2018). Automatic chinese short answer grading with deep autoencoder. In *International Conference on Artificial Intelligence in Education*, p. 399–404 : Springer.