Actes de la 6e conférence conjointe Journées d'Études sur la Parole (JEP, 33e édition),
Traitement Automatique des Langues Naturelles (TALN, 27e édition),
Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RÉCITAL, 22e édition)
Nancy, France, 08-19 juin 2020

Recherche de similarité thématique en temps réel au sein d'un débat en ligne

Mathieu Lafourcade[1, 2] Noémie-Fleur Sandillon-Rezer[1, 2]
(1) LIRMM, 161 rue Ada, 34095 Montpellier Cedex 5, France
(2) Université de Montpellier, 163 rue Auguste Broussonnet, 34090 Montpellier, France
mathieu.lafourcade@lirmm.fr, noemie-fleur.sandillon-rezer@lirmm.fr

RÉSUMÉ

Cet article se focalise sur l'utilisation d'un large réseau lexico-sémantique français pour le calcul de similarité thématique d'interventions au cours d'un débat en ligne dans les lycées, proche du temps réel. Pour cela, notre système extrait des informations sémantiques du réseau et crée à la volée des vecteurs enrichis pour chaque fragment de texte. Les données récupérées sont contextualisées via un algorithme de propagation. Les vecteurs résultat permettent aux fragments de texte d'être comparés. Notre méthode aide à trouver les thématiques émergentes des débats et à identifier des clusters d'opinion. La contrainte temps réel nous force à sélectionner précisément les informations que nous incluons, aussi bien pour les temps de calcul des vecteurs créés que la qualité de ceux-ci.

ABSTRACT

Thematic similarity real-time computation during an online debate

This paper describes the use of a large French lexical and semantic network for text embedding computation for thematic similarity, as close as possible to real time, in the context of in-school online debates. To this purpose, our system creates on the fly enriched vectors that embed thematic aspects of text fragments. Semantic information associated to textual contents are retrieved from a knowledge base, then contextualised by a graph propagation algorithm. Those lexicalized vectors allow texts to be thematically compared. The system helps teachers by finding emergent topics of discussions or identifying clusters of opinions. The real-time constraint forces us to choose precisely which semantic processing we include in vector building, as they can have a crucial impact.

MOTS-CLÉS : proximité thématique, réseau lexico-sémantique, vecteurs lexicalisés.

KEYWORDS: thematic similarity, lexical and semantic network, lexicalized vectors.

1 Introduction

Cet article présente une utilisation d'un réseau lexico-sémantique français pour le calcul de vecteurs textuels enrichis, cela dans le but de trouver des similarités thématiques entre des fragments de texte. Nous nous plaçons dans le cadre du projet AREN (ARgumentation Et Numérique), où notre méthode est appliquée en temps réel à des débats en ligne entre lycéens. Ce projet, soutenu par le ministère de l'Éducation Nationale, est un des lauréats de l'appel e-FRAN [1]. Son but principal est d'apprendre les mécanismes du débat aux lycéens via une plateforme mise en ligne et utilisée depuis

1. Espaces de Formation, de Recherche et d'Animation Numérique

Volume 2 : Traitement Automatique des Langues Naturelles, pages 258–267.
HAL : hal-02784775.

2017 (voir figure 1). Un des objectifs secondaires est d'exploiter les techniques du TAL pour assister élèves et enseignants durant un débat ou pendant l'étape de restitution. En effet, un débat en classe (dans le cadre du projet) se divise en trois parties : la préparation en classe, où les élèves acquièrent des données et connaissances sur le sujet à débattre, le débat en ligne, d'une durée de 50 minutes en général et la restitution du débat. Lors de celle-ci, un travail consiste à résumer les différents arguments du débat. Pour cela, pouvoir rassembler les arguments par thème est d'une grande aide pour les enseignants, leur permettant de vérifier ce que propose le système plutôt que lire parfois jusqu'à 300 arguments pour commencer le tri. Durant le débat, cela peut également permettre à l'enseignant de mettre l'accent sur un thème qu'il souhaiterait voir abordé et qui n'a pas encore assez de contributions. Les textes analysés sont donc courts (généralement une phrase), et écrits sur le vif.

L'idée soutenant notre méthode est de créer des vecteurs pour chaque contribution textuelle, d'enrichir ceux-ci via le réseau lexico-sémantique et de les comparer par un produit scalaire. Il est ainsi possible de déterminer automatiquement quelles contributions sont proches thématiquement les unes des autres, et ce même si le vocabulaire utilisé diverge.

FIGURE 1 – Interface d'AREN : texte débattu à gauche, commentaires à droite découpés en 3 parties : sélection (extrait à commenter), reformulation (de la sélection), argumentation (où on s'exprime).

Il est important de garder à l'esprit deux aspects du projet. La contrainte temps-réel nous impose d'utiliser des approches rapides et nous ne vérifions pas que les interventions sont sémantiquement convergentes (les négations, par exemple, ne sont pas prises en compte), mais simplement qu'elles portent sur le même thème. En outre, les vecteurs sont lexicalisés (ensemble de paires mot-poids) pour une interprétation humaine et machine facilitée, dans l'esprit de Panigrahi *et al.* (2019). Les vecteurs sont en dimension ouverte. Ils peuvent être composés d'autant de paires mot-poids que souhaité. Pour être comparés deux à deux, le fait de ne pas avoir la même dimension n'est pas problématique.

Les données utilisées sont les débats réalisés au cours des 4 ans de projet. Celles-ci seront mises à disposition au terme du projet, ainsi que le code produisant les vecteurs et la plateforme utilisée, le tout sous licence libre.

Dans cet article, nous commencerons par décrire rapidement certains aspects de la base de connaissance JeuxDeMots dont nous nous servons pour l'augmentation sémantique. Après un rapide aperçu des méthodes récentes sur les plongements de mots et de textes, nous décrirons notre méthodologie, puis l'évaluerons avant de conclure.

2 Utiliser une grande base de connaissances lexicalisées

Le projet JeuxDeMots [2] (Lafourcade, 2007) (JDM) a pour cœur des GWAP (un jeu en ligne, voir Ahn (2006)) où des joueurs s'affrontent pour capturer des mots, combiné à des mécanismes d'inférences. Le ressort principal (Lafourcade *et al.*, 2018) est de leur faire produire des associations entre termes selon une consigne (par exemple : donner des synonymes de "chat"). Ainsi, le réseau lexical JDM, dont la structure est composée de nœuds connectés par des relations (voir Collins & Quillian (1969), Sowa & Zachman (1992), Gaume *et al.* (2007) et Polguère (2014)), se développe en fonction de l'activité des participants (environ 4 millions de termes et 310 millions de relations en janvier 2020). Il existe environ 120 types de relations pouvant être organisés selon les catégories suivantes :

Relations lexicales - Focalisées sur le vocabulaire et la lexicalisation, cela correspond à la synonymie, antonymie, champ lexical, etc.
Relations ontologiques - Centrées sur les connaissances de la langue, il s'agit des génériques (hyperonymie), spécifiques (hyponymie), parties de (meronymie), lieux spécifiques, etc.
Relations associatives - Plus subjectives, elle font appel à la culture générale : associations libres, sentiments associés, gloses, objets similaires, objets souvent présents ensembles.
Relations prédicatives - Associées à un vembe ou un nom d'action, aussi bien qu'aux valeurs des arguments : agent, patient, lieux où une action se déroule, etc.
Sens des termes permettant de représenter des raffinements spécifiques, par exemple : frégate>navire de frégate>oiseau.

En plus d'être typée, une relation est pondérée, le poids pouvant être négatif, indiquant une relation fausse ou impossible. Enfin, les relations peuvent être annotées de façon ouverte avec diverses informations : fréquences, pertinence, prépositions (pour les relations de lieux), subjectivité, etc. La base est très lexicalisée, en ce sens que les verbes arrivent avec leur formes conjuguées ; les groupes nominaux avec leur pluriel, etc. Un grand nombre de formes verbales infinitives arrivent avec leur version négative (manger / ne pas manger / ne plus manger, etc.) et leurs associations sémantiques respectives. Cela résulte d'un ajout de contributeurs hors jeu, et permet lors d'une analyse sémantique de rejeter certains sens.

3 Calcul de vecteurs avec une large base de connaissances

Contrairement aux approches récentes, où les plongements textuels sont calculés à partir d'un grand corpus – souvent Wikipedia Ein Dor *et al.* (2018)–, on s'appuie sur une large base de connaissances.

Les techniques de fouilles textuelles cherchant à extraire des données pertinentes de texte sont souvent utilisées pour mesurer la similarité (Vijaymeena1 & Kavitha (2016), Sumathy & Chidambaram (2016), Peinelt *et al.* (2019) et Gong *et al.* (2018)). L'enjeu des modèles à base de vecteurs est de construire lesdits vecteurs. En contrepartie, on adopte généralement ces modèles pour la facilité de comparaison des vecteurs. On peut citer à la volée : les méthodes LSA - Latent Semantic Analysis - (Magerman *et al.*, 2011), LDA - Latent Dirichlet Allocation - (Liu *et al.*, 2015), LSTM - Siamese Long Short Term Memory - (Melamud *et al.*, 2016) ou encore Doc2Vec, foncé sur Word2Vec (Le & Mikolov, 2014). Les réseaux neuronaux convolutionnels commencent également à être utilisés (Zheng *et al.*, 2019). Généralement, les approches sont entraînées sur larges corpus, vu qu'il est délicat de créer des

2. http://www.jeuxdemots.org

vecteurs pertinents sans aucune source de connaissance (Park *et al.*, 2018). Cela rend la récupération des informations sous-entendues plus complexe (Smalheiser *et al.*, 2019) , alors qu'un lecteur humain le fait aisément tant que lesdites informations viennent de son environnement culturel. Nous avons délaissé les méthodes telles que ELMo (Peters *et al.*, 2018) BERT (Devlin *et al.*, 2019) car nous souhaitions une méthodologie en contrôle, où nous choisissons les relations sémantiques où sont puisés les termes à ajouter.

Actuellement, nous exploitons la proximité thématique (par produit scalaire entre vecteurs) selon deux circonstances : (a) vérifier automatiquement que la reformulation est (thématiquement) similaire à la sélection (leurs vecteurs ont un produit scalaire élevé) et (b) identifier les commentaires (thématiquement) proches d'un groupe de mot pouvant jouer le rôle de thème spécifique. Cela permet de recentrer le débat facilement, de vérifier si certaines idées sont exprimées et combien de fois.

3.1 Schéma général de construction de vecteurs lexicalisés

L'idée maîtresse est de déclarer différentes étapes et de construire un pipeline avec. On différencie les étapes sémantiques – qui font appel aux connaissances issues de JeuxDeMots et ajoutent des informations importantes, comme les synonymes (Abdalgader & Skabar, 2011), et améliorent la qualité des vecteurs (Espinosa Anke *et al.*, 2019), avec les co-locations – et celles qui ne le sont pas (étapes de régularisation), mais qui sont indispensables pour construire des vecteurs (où les mots sont associés à leur poids). Le pipeline complet est illustré figure 2. Nos choix se fondent sur un compromis entre de temps de calcul et qualité. Il nous a bien sûr fallu faire des choix, principalement en terme d'augmentation, pour rester dans l'optique du temps réel (un débat en classe dure 50 minutes, il est nécessaire que les calculs soient terminés à la fin de celui-ci pour faire le point avec les élèves).

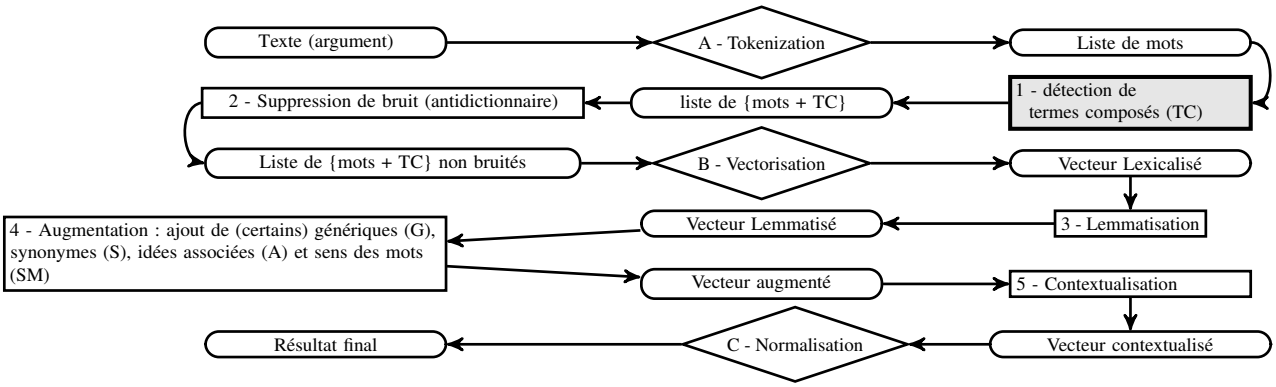

FIGURE 2 – Processus global ; les étapes sémantiques (1 - 5) exploitent la base de connaissances.

3.2 Étapes de régularisation

Elles peuvent être utilisées seules pour obtenir des vecteurs élémentaire très rapidement. Ces vecteurs peuvent être utilisés comme références (baseline). On considère un fragment de texte (ou fragment) comme une suite de mots $w_1 s_1 w_2 s_2 ... s_k w_n$ ($n, k \in \mathbb{N}$), où w_i représente un mot et s_j un séparateur.

A - Tokenisation : transforme un fragment en tokens, en détectant les séparateurs : $(w_1, w_2, ..., w_n)$.
B - Vectorisation : crée un vecteur en comptant et fusionnant les tokens dupliqués. L'ensemble pondéré obtenu est appelé vecteur lexicalisé : $\{w_1 : 2, w_2 : 1, ..., w_{n'} : 3\}$ où $n' \in \mathbb{N}$ et $n' \leq n$.
C - Normalisation : modifie les poids avec une norme euclidienne. Soit un vecteur (x, y), une fois normé il devient $(\frac{x}{\sqrt{x^2+y^2}}, \frac{y}{\sqrt{x^2+y^2}})$). Cela permet la comparaison de deux vecteurs.

3.3 Étapes sémantiques

Ces étapes sémantiques ajoutent des informations sémantiques au vecteur, en atténuant les fluctuations lexicales de langage utilisé, et surtout en identifiant les concepts sous-tendus tels que identifiés a priori par les locuteurs (dans le cadre du projet JeuxdeMots) aussi bien qu'en réduisant l'ambiguïté en identifiant les termes composés.

1 - Détection des termes composés (TC) : est nécessaire pour éviter les erreurs d'interprétation aussi bien que pour le raffinement. Par exemple *île flottante* est plus certainement un dessert qu'une île qui flotte et *véhicule autonome* est un concept précis qui a un sens et un champ lexical bien à lui.

Les termes composés sont extraits sous forme de liste de JDM ; celle-ci permet de créer un automate à états finis, qui lira les fragments (en prenant en compte les séparateurs) et concatènera les séquences correspondant à un TC avec "_" : *véhicule autonome* deviendra *véhicule_autonome*. Lorsque deux TCs se superposent, nous sélectionnons le terme le plus à droite (heuristique correspondant à ce qu'on observe plus souvent en français). La détection de TCs, pour être complète, doit être effectuée en trois passes, accompagnée entre chaque d'une *lemmatisation progressive*. Cela demande de construire le graphe des lemmatisations possibles et devient coûteux en temps. Les trois passes sont donc :
Sans lemmatisation : couvre les cas où les traits grammaticaux comptent (*monteur de câbles d'avions*).
Lemmatisation verbale : pour reconnaître des expressions telles que *mettre les pieds dans le plat*.
Lemmatisation totale : détecte les TCs enregistrés sous forme lemmatique, tel que *véhicule autonome*.

2 - Retrait du bruit : à partir d'une liste de mots vides – du bruit – on crée une expression rationnelle qui retire ces mots des fragments.

3 - Lemmatisation : récupère les lemmes des mots, depuis JDM. Leur poids est le même que celui du mot initial. Lorsque l'on effectue la détection des termes composés complète, cette étape y est incluse. Cependant, pour une exécution plus rapide, on peut réduire la détection des termes composés à sa première étape et il est alors nécessaire d'effectuer la lemmatisation à part.

4 - Augmentation : demande à JDM les mots associés (génériques, synonymes, idées associées et sens des mots) et récupère une liste de termes pondérés. Cette liste est triée par poids de relation, les négatifs sont exclus, et nous récupérons les k premiers, qui sont ajoutés au vecteur avec un facteur d'atténuation f qui s'applique sur le poids du terme initial. Le facteur d'atténuation ainsi que le nombre de termes sélectionnés peuvent être modifiés à l'appréciation de l'utilisateur. Empiriquement et dans le cadre de l'utilisation que nous faisons de la similarité thématique dans le projet AREN, nous avons fixé $f = 0.8$, pour les synonymes $k = 10$ et pour les autres augmentations $n = 3$. Si les résultats existent déjà dans le vecteur, les poids se cumulent.

5 - Contextualisation : a pour but de garder les termes ajoutés les plus pertinents, en fonction du contexte textuel – Chapuis & Lafourcade (2017) pour une approche similaire –. L'algorithme prend en entrée le vecteur construit jusqu'à l'étape 4, et le considère comme un ensemble de termes pondérés $S = \{t_1/w_1 ; ...t_n/w_n\}$. On normalise les poids de manière à ce que le plus élevé soit égal à 1 ; on construit S' de la même manière. On crée un graphe en ajoutant toutes les relations possibles entre les éléments de $S \cup S'$ trouvées dans JDM (celles-ci peuvent être positives, donc vraies, ou négatives, fausses). On passe à la phase d'initialisation. À chaque terme de départ (de S), on assigne une valeur d'activation a égale à son poids dans S. Les termes de S' ont une valeur d'activation de 0. Vient ensuite l'étape de propagation : on propage la valeur d'activation de chaque nœud N à chaque voisin de S', via la relation R d'un poids $w(R)$ tel que $a(S'_i) \leftarrow (a(S'_i) \times a(N) \times w(R))^{1/3}$. Un nœud N est donc considéré comme un neurone qui transmet son activation $a(N)$ si celle-ci est au dessus

d'un certain seuil (empiriquement, ce seuil est de 0.5). L'étape suivante, d'itération, ajoute à la valeur d'activation de chaque terme initial (de S) son poids dans S, avant de repasser à l'étape de propagation. On répète jusqu'à convergence des poids ou jusqu'au maximum d'étapes autorisées. L'algorithme est prouvé non convergent en général, mais converge dans les cas où il n'y a pas d'interprétations multiples au texte d'entrée. En contextualisant notre exemple, on obtient :

Rafinements gardés	**Rafinements écartés**	voiture>automobile>jouet: -284
voiture>automobile : 228	voiture>train: -182	piéton>facteur: -333
piéton>personne se déplaçant à pied: 98	voiture>véhicule de transport à roues: -204	piéton>soldat: -333
danger>péril: 74	voiture>mode de transport: -284	danger>marine: -343
		danger>inconvénient: -343

Par exemple, on trouve dans le graphe le chemin *voiture>automobile* → *voiture autonome* → *danger>péril*, ce qui permet de renforcer ces sélections. Plus des mots ont des liens, plus ils sont renforcés et sont susceptibles d'être identifiés comme les sens les plus probables. Cependant, cette méthode est dépendante des informations présentes dans la base de connaissances.

3.4 Exemple

En partant de la phrase "les voitures autonomes sont un danger pour les piétons" et en y appliquant toutes les étapes décrites ci-dessus nous obtenons :

```
{voitures_autonomes:0.25, automobile:0.25, voiture_autonome:0.25, danger:0.25
;voiture>automobile:0.25, voiture:0.25, être un danger:0.25, piéton>personne se
déplaçant à pied:0.25, risque:0.17, inquiétude:0.15, menace:0.12, difficulté:0.12,
piéton:0.25, passant:0.23, marcheur:0.23, individu:0.12 personne:0.12;
véhicule:0.17, véhicule terrestre:0.23, accident de la circulation:0.95 ;écraser un
piéton:0.23, renverser un piéton:0.23, accident:0.19, rue:0.16
```

On notera que les termes composés sont reconnus comme tels (*voitures autonomes* et *être un danger*) et que, à l'aide de la contextualisation, le verbe *piéter* a été écarté comme lemme pour *piétons*.

4 Évaluation et discussion

Les données du projet, correspondant à 6481 arguments de 77 débats, nous ont permis d'évaluer notre méthode : en premier lieu, au niveau des temps de calcul des vecteurs, sur un ordinateur personnel (16 GO de RAM, processeur 2,2 GHz Intel Core i7 quatre cœurs). Avec la baseline (étapes de régularisation), il faut **0.339s** pour calculer les 6481 vecteurs. En ajoutant les étapes sémantiques, on a au total **5159.42s**, moyen **0.79s**, minimal **4 10^{-5}s** et maximal **6.44s** (correspond à une intervention très longue (355 mots), particulièrement rare lors des débats). Les points les plus coûteux sont les appels à la base de connaissance et la contextualisation (**0.58s**). La base de connaissance étant codée sous forme de base de données indexée, sa taille n'a que peu d'influence sur la complexité, c'est le nombre de requêtes effectuées qui joue.

L'évaluation qualitative sans corpus de référence est délicate. Nous avons donc créé notre propre Gold Standard : pour chaque débat et chaque argument du débat, nous avons ordonné les 5 plus proches. Il est donc ensuite possible de comparer avec les résultats de notre méthode mais également avec les vecteurs calculés de façon alternative, ici en particulier selon la méthode de Bojanowski *et al.* (2017) qui utilise FastText (dimension de 300 et modèle skip-gram). Après adaptation (somme vectorielle normée des vecteurs de chaque terme des segments textuels, termes pour lesquels un vecteur existe), nous avons pu l'appliquer à nos fragments textuels et comparer les résultats selon les approches.

Méthode de construction	Précision/GSM	Gain / MSs	Gain / Sans CT
Mots Simples (MSs)	42.08%		
MSs + Termes Composés (TCs)	67.83%	+25.7	
MSs + TCs + Lemmatisation (L)	78.41%	+36.2	
MSs + TCs + L + Génériques (G)	84.23%	+42.3	
MSs + TCs + L + Synonymes (S)	85.71%	+43.6	
MSs + TCs + L + G + S + Termes associés	91.31%	+49.2	
Mots Simples (MSs) + Contextualisation (CT)	92.44%		+50.2
MSs + TCs + CT	97.08%	+4.7	+29.3
MSs + TCs + Lemmatisation (L) + CT	98.31%	+6	+21.1
MSs + TCs + L + Génériques (G) + CT	98.67%	+6.3	+14.3
MSs + TCs + L + G + CT	98.64%	+6.2	+13.1
MSs + TCs + L + G + S + Sens des mots + termes associés + CT	99.92%	+7.5	+8.6
MSs + Bojanowski *et al.*	45.21%		
MSs + TCs + L + G + S + B. *et al.*	56.71%		
PWs + TCs + L + G + S + Sens des mots + A + CT + B. *et al.*	61.34%		

TABLE 1 – Pourcentage d'arguments ordonnés correctement par rapport à notre Gold Standard manuel (GSM), en fonction des différentes méthodes utilisées.

La table 1 montre la précision des calculs pour chaque pipeline et il apparaît clairement que l'augmentation et la contextualisation ont un effet très positif sur la qualité des vecteurs (et donc du rapprochement thématique). Les cas d'échec ont systématiquement été identifiés comme des relations manquantes dans la base de connaissance, qui peut être complétée de façon rapide. Enfin, la forte lexicalisation de la base induit que le rapprochement est souvent sémantique, en particulier selon l'usage ou non de la négation (notamment, car les formes verbales négatives sont présentes dans la base). Par exemple, les segments "une voiture autonome peut ne pas polluer", "la voiture autonome est verte" et "la voiture intelligente est écologique" sont identifiés comme très proches.

Nous avons également comparé notre approche avec de la méthode de Bojanowski *et al.*, dont les résultats sont assez proches de notre baseline, ce qu'on peut expliquer par le fait que le plongement de termes isolés n'est pas équivalent à l'augmentation + contextualisation. Par ailleurs, il est possible que l'entraînement en utilisant Wikipédia ne soit pas adéquat pour un débat en ligne, où les arguments sont assez spontanés.

5 Conclusion et travail futur

En nous écartant des méthodes actuelles de plongements de termes, nous avons mis au point un calcul de vecteurs à la volée qui s'appuie fortement sur la base de connaissances JeuxDeMots ainsi qu'une procédure de contextualisation thématique. Nos résultats sont adaptables et explicables, et exhibe une ambiguïté sémantique réduite. Les ressources machines nécessaires pour les calculs sont raisonnables (l'ensemble tourne sur un ordinateur de bureau classique) et nous sommes proches du temps réel (sous la seconde), ce qui était requis. Nous avons effectivement testé notre méthode dans les classes lors du projet AREN, et obtenus des retours très positifs sur la qualité des rapprochements thématiques obtenus. De plus, si le corpus d'arguments utilisé pour tester notre méthode n'est actuellement pas disponible au public, il sera anonymisé et rendu disponible à la fin du projet.

Bien que n'étant pas un objectif premier, le graphe de relations produit par la contextualisation est lisible par un être humain et peut être utilisé pour expliquer le résultat exprimé ; fonctionnalité d'explication qui sera ajoutée dans la plate-forme. Nous comptons également améliorer plus avant notre précision, en testant l'exploitation d'autres informations de JeuxDeMots et élargir notre cercle d'utilisateurs à la société civile pour construire un Gold Standard plus large et librement accessible.

Références

ABDALGADER K. & SKABAR A. (2011). Short-text similarity measurement using word sense disambiguation and synonym expansion. In J. LI, Éd., *AI 2010 : Advances in Artificial Intelligence*, p. 435–444, Berlin, Heidelberg : Springer Berlin Heidelberg.

AHN L. V. (2006). Games with a purpose. *Computer*, **39**(6), 92–94. DOI : 10.1109/MC.2006.196.

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.

CHAPUIS M. & LAFOURCADE M. (2017). Identifying Polysemous Words and Inferring Sense Glosses in a Semantic Network. In *IWCS : International Conference on Computational Semantics*, Montpellier, France. HAL : lirmm-01763423.

COLLINS A. M. & QUILLIAN M. R. (1969). Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behavior*, **8**(2), 240 – 247. DOI : 10.1016/S0022-5371(69)80069-1.

DEVLIN J., CHANG M.-W., LEE K. & TOUTANOVA K. (2019). BERT : Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers)*, p. 4171–4186, Minneapolis, Minnesota : Association for Computational Linguistics. DOI : 10.18653/v1/N19-1423.

EIN DOR L., MASS Y., HALFON A., VENEZIAN E., SHNAYDERMAN I., AHARONOV R. & SLONIM N. (2018). Learning thematic similarity metric from article sections using triplet networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers)*, p. 49–54, Melbourne, Australia : Association for Computational Linguistics. DOI : 10.18653/v1/P18-2009.

ESPINOSA ANKE L., SCHOCKAERT S. & WANNER L. (2019). Collocation classification with unsupervised relation vectors. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, p. 5765–5772, Florence, Italy : Association for Computational Linguistics. DOI : 10.18653/v1/P19-1576.

GAUME B., DUVIGNAU K. & VANHOVE M. (2007). Semantic associations and confluences in paradigmatic networks. In M. VANHOVE, Éd., *From polysemy to semantic change - towards a typology of lexical semantic associations* : John Benjamins Publishing Company. DOI : 10.1075/slcs.106.11gau, HAL : hal-01321894.

GONG H., SAKAKINI T., BHAT S. & XIONG J. (2018). Document similarity for texts of varying lengths via hidden topics. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 2341–2351, Melbourne, Australia : Association for Computational Linguistics. DOI : 10.18653/v1/P18-1218.

LAFOURCADE M. (2007). Making people play for Lexical Acquisition with the JeuxDeMots prototype. In *SNLP'07 : 7th International Symposium on Natural Language Processing*, Pattaya, Chonburi, Thailand. HAL : lirmm-00200883.

LAFOURCADE M., MERY B., MIRZAPOUR M., MOOT R. & RETORÉ C. (2018). Collecting Weighted Coercions from Crowd-Sourced Lexical Data for Compositional Semantic Analysis. In *isAI : International Symposium on Artificial Intelligence*, volume LNCS de *New Frontiers in Artificial Intelligence*, p. 214–230, Tokyo, Japan. DOI : 10.1007/978-3-319-93794-6_15, HAL : lirmm-01916209.

LE Q. V. & MIKOLOV T. (2014). Distributed Representations of Sentences and Documents. *arXiv e-prints*, p. arXiv :1405.4053. arXiv : 1405.4053.

LIU Y., LIU Z., CHUA T.-S. & SUN M. (2015). Topical word embeddings. In *Proceedings AAAI Conference on Artificial Intelligence*.

MAGERMAN T., VAN LOOY B., BAESENS B. & DEBACKERE K. (2011). Assessment of latent semantic analysis (lsa) text mining algorithms for large scale mapping of patent and scientific publication documents. *Katholieke Universiteit Leuven Department of Mangerial Economics Strategy and Innovation, Working Paper 1114*, p. 1–7.

MELAMUD O., GOLDBERGER J. & DAGAN I. (2016). context2vec : Learning generic context embedding with bidirectional LSTM. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, p. 51–61, Berlin, Germany : Association for Computational Linguistics. DOI : 10.18653/v1/K16-1006.

PANIGRAHI A., SIMHADRI H. V. & BHATTACHARYYA C. (2019). Word2Sense : Sparse interpretable word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, p. 5692–5705, Florence, Italy : Association for Computational Linguistics. DOI : 10.18653/v1/P19-1570.

PARK S., BYUN J., BAEK S., CHO Y. & OH A. (2018). Subword-level word vector representations for Korean. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 2429–2438, Melbourne, Australia : Association for Computational Linguistics. DOI : 10.18653/v1/P18-1226.

PEINELT N., LIAKATA M. & NGUYEN D. (2019). Aiming beyond the obvious : Identifying non-obvious cases in semantic similarity datasets. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, p. 2792–2798, Florence, Italy : Association for Computational Linguistics. DOI : 10.18653/v1/P19-1268.

PETERS M., NEUMANN M., IYYER M., GARDNER M., CLARK C., LEE K. & ZETTLEMOYER L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long Papers)*, p. 2227–2237, New Orleans, Louisiana : Association for Computational Linguistics. DOI : 10.18653/v1/N18-1202.

POLGUÈRE A. (2014). From writing dictionaries to weaving lexical networks. *International Journal of Lexicography*, **27**(4), 396—418.

SMALHEISER N. R., COHEN A. M. & BONIFIELD G. (2019). Unsupervised low-dimensional vector representations for words, phrases and text that are transparent, scalable, and produce similarity metrics that are not redundant with neural embeddings. *Journal of Biomedical Informatics*, **90**, 103096. DOI : https ://doi.org/10.1016/j.jbi.2019.103096.

SOWA J. F. & ZACHMAN J. A. (1992). Extending and formalizing the framework for information systems architecture. *IBM Systems Journal*, **31**(3), 590–616. DOI : 10.1147/sj.313.0590.

SUMATHY M. K. L. & CHIDAMBARAM D. (2016). A hybrid approach for measuring semantic similarity between documents and its application in mining the knowledge repositories. *International Journal of Advanced Computer Science and Applications*, **7**(8).

THONGTAN T. & PHIENTHRAKUL T. (2019). Sentiment classification using document embeddings trained with cosine similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics : Student Research Workshop*, p. 407–414, Florence, Italy : Association for Computational Linguistics. DOI : 10.18653/v1/P19-2057.

VIJAYMEENA1 M. & KAVITHA K. (2016). A survey on similarity measures in text mining. *Machine Learning and Applications, Machine Learning and Applications : An International Journal (MLAIJ)*, **3(1)**, 19–28.

ZHENG T., GAO Y., WANG F., FAN C., FU X., LI M., ZHANG Y., ZHANG S. & MA H. (2019). Detection of medical text semantic similarity based on convolutional neural network. *BMC Medical Informatics and Decision Making*, **19**(1), 156. DOI : 10.1186/s12911-019-0880-2.