Représentation vectorielle de paires de verbes pour la prédiction de relations lexicales

Etienne Rigaud

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

Représentation vectorielle de paires de verbes pour la prédiction de relations lexicales

Etienne Rigaud[1, 2]
(1) LORIA, 615 Rue du Jardin-Botanique, 54506 Vandœuvre-lès-Nancy
(2) École des Mines de Nancy, 54000 Nancy
etienne.rigaud6@etu.univ-lorraine.fr

RÉSUMÉ

Dans cet article, nous proposons un modèle de représentations vectorielles de paire de mots, obtenues à partir d'une adaptation du modèle Skip-gram de Word2vec. Ce modèle est utilisé pour générer des vecteurs de paires de verbes, entraînées sur le corpus de textes anglais Ukwac. Les vecteurs sont évalués sur les données ConceptNet & EACL, sur une tâche de classification de relations lexicales. Nous comparons les résultats obtenus avec les vecteurs paires à des modèles utilisant des vecteurs mots, et testons l'évaluation avec des verbes dans leur forme originale et dans leur forme lemmatisée. Enfin, nous présentons des expériences où ces vecteurs paires sont utilisés sur une tâche d'identification de relation discursive entre deux segments de texte. Nos résultats sur le corpus anglais *Penn Discourse Treebank*, démontrent l'importance de l'information verbale pour la tâche, et la complémentarité de ces vecteurs paires avec les connecteurs discursifs des relations.

ABSTRACT

Verb-pairs embeddings for discourse relation prediction

This paper proposes a model to obtain vector representations of pairs of words, obtained from an adaptation of the Word2vec Skip-gram Model. This model is used to generate embeddings for pairs of verbs, trained on the english corpus Ukwac. The pair-embeddings are then evaluated on a classification task, where the goal is to predict the lexical relation between the input pair of words. The scores obtained on this task with the pair-embeddings are compared with the scores obtained with individual word-embeddings and with pairs of lemmatized verbs. Finally, the pair-embeddings are used on the discourse relation prediction task, on the *Penn Discourse Treebank* dataset, revealing the relevance of verbs for this task, and the complementarity between the verbs and the discourse connective.

MOTS-CLÉS : Vecteur mot, vecteur relation de verbes, analyse de texte, prédiction de relation du discours.

KEYWORDS: Word embedding, relation embedding between verbs, text analysis, discourse relation prediction.

1 Introduction

Dans le domaine du traitement automatique des langues, l'apprentissage de relations entre deux mots donnés est extrêmement important, pour des tâches telles que répondre à des questions, la recherche d'antonymes ou de synonymes, ou bien simplement pour mieux comprendre comment les langages se construisent. Cela motive les recherches présentées dans cet article, qui ont pour

Volume 3 : Rencontre des Étudiants Chercheurs en Informatique pour le TAL, pages 179–192.
HAL : hal-02786199.

objectif de construire des représentations de relations entre une paire de mots, et d'évaluer la qualité de ces représentations dans différentes tâches. Pour ce faire, nous présentons un modèle permettant d'entraîner des représentations vectorielles de relations entre des paires de mots.

Les représentations vectorielles de mots sont très utilisées en traitement automatique des langues depuis les années 1990, l'idée étant que les mots sémantiquement proches se retrouvent proches dans l'espace vectoriel des mots. On retrouve souvent dans la littérature le terme *word embedding* ou "plongement lexical" pour qualifier ces représentations vectorielles. Ces représentations sont fondées sur l'hypothèse distributionnelle (Harris, 1954) : "les lexèmes possédant un contexte linguistique similaire ont un sens similaire.". On peut par exemple obtenir des représentations vectorielles de mots en construisant une matrice de co-occurrences, en comptant le nombre d'occurrences de mots dans le contexte d'autres mots (Turney & Pantel, 2010). L'inconveniant de cette méthode étant néanmmoins la taille importante des vecteurs obtenus, et leurs composantes qui sont majoritairement nulles.

Outre ce modèle distributionnel, les modèles utilisant des réseaux de neurones se sont fortement développés, un des modèles populaires étant Word2vec (Mikolov *et al.*, 2013a). Ce modèle consiste à entraîner une matrice de plongements de mots, qui se trouve être une des matrices de poids du réseau, à travers une tâche précise. Le modèle Skip-gram de Word2vec prend un mot en entrée et cherche à prédire la probabilité qu'a un autre mot d'apparaître dans le contexte du mot donné en entrée. Le modèle utilisé dans cet article est une adaptation du modèle Skip-gram à des paires de mots, qui génère des *embeddings* pour des paires de mots.

Dans un premier temps, nous présentons les précédents travaux sur la représentation de relations sémantiques, puis développons une approche particulière de représentation, appliquée aux relations entre des paires de verbes en anglais. Enfin, les expériences menées ainsi que les résultats sont détaillés pour comparer la qualité de ce modèle à d'autres approches. Le code et les données seront rendus disponibles à la communauté.

2 État de l'art

L'apprentissage de représentations vectorielles pour des relations lexicales est très utile pour de nombreuses tâches, comme l'inférence, la recherche d'analogie ou la classification de relations du discours (Braud & Denis, 2016). Des modèles basés sur la composition de vecteurs mots permettent d'apprendre des relations complexes. En effet, ces vecteurs vérifient des relations algébriques qui traduisent des relations sémantiques. Par exemple $\vec{v}_{France} - \vec{v}_{Paris} + \vec{v}_{Madrid} \approx \vec{v}_{Espagne}$ (Mikolov *et al.*, 2013b).

De cette équation, on peut déduire que $\vec{v}_{France} - \vec{v}_{Paris} \approx \vec{v}_{Espagne} - v_{Madrid} \approx \vec{v}_{est\,capitale\,de}$.

Des travaux récents montrent des modèles de représentations de relations entre plusieurs mots, voire de *sentence embeddings* fonctionnant en composant les vecteurs de chaque mot de la phrase (Weston *et al.*, 2013; Hill *et al.*, 2016). Les vecteurs relations peuvent être appris en adaptant les modèles d'apprentissage de vecteur mot Word2vec, comme cela a été fait dans Chingacham & Paperno (2018) : ces auteurs proposent une adaptation de Skip-gram aux paires de noms, et en entraînant un réseau de mapping avec des vecteurs mots pré-entraînés pour régler le problème des données éparses. Similairement, Jameel *et al.* (2018) ont adapté le modèle GloVe (Pennington *et al.*, 2014) pour apprendre des représentations de relations entre une paire de mot, sans l'utilisation de vecteurs mots pré-entraînés.

Les travaux récents utilisent en grande majorité de l'apprentissage non supervisé pour apprendre les représentations de relations, comme c'est le cas avec les modèles adaptant Word2vec (Joshi *et al.*, 2018; Chingacham & Paperno, 2018). Ces modèles sont en général transférés par la suite sur d'autres tâches, comme de l'inférence ou de l'annotation d'image par exemple. Des recherches récentes ont exploré différents modèles possibles pour entraîner des représentations de relations sémantiques, en cherchant à obtenir les meilleures performances possibles sur ces tâches de transfert. Les modèles de réseaux récurrents *LSTM* ou *LSTM Bi-directionnel* obtiennent par exemple de très bons résultats sur une grande variété de tâches. La difficulté réside dans le choix de l'objectif de ces modèles, afin de créer des vecteurs capturant le maximum d'information sur une unité de texte. De très bons résultats ont ainsi été obtenus en entraînant des réseaux de neurones complexes à faire de l'inférence de textes (Conneau *et al.*, 2017), de la classification de relations du discours explicites (Nie *et al.*, 2019) ou bien de la prédiction de marqueurs du discours (Sileo *et al.*, 2019). Néanmoins, ces modèles complexes nécessitent d'immenses quantités de données, ainsi qu'une très importante puissance de calcul. Comme dans ces approches, nous cherchons à construire une représentation en utilisant une architecture neuronale, mais nous nous distinguons par la simplicité du modèle utilisé.

3 Approche

En général, les représentations sont construites pour un mot ou un fragment textuel comme une phrase ou pour un document entier. Notre approche consiste à construire une représentation pour une paire de mots non adjacents apparaissant dans une phrase : nous espérons ainsi construire une représentation de la relation qui lie les mots considérés. Cette représentation doit prendre en compte le contexte d'apparition de la paire de mots. Nous détaillerons dans le reste de cette section le modèle utilisé pour obtenir des représentations vectorielles de relations entre une paire de verbes. Ce modèle est adapté du travail de A. Chingacham (Chingacham & Paperno, 2018) sur les paires de noms.

3.1 Modèle Skip-gram original

Le modèle Skip-gram est un réseau de neurones à une couche cachée, entraîné à prédire le contexte dans une certaine fenêtre d'un mot donné. Considérons l'exemple ci-dessous, extrait du corpus anglais Ukwac (Ferraresi *et al.*, 2013), avec une taille de fenêtre pour le contexte égale à 5.

Exemple 1 *Sensible guidelines will need to be established for holding case details at a regional level.*

Dans ce modèle, la matrice de poids entre la couche d'entrée et la couche cachée évolue jusqu'à contenir des représentations vectorielles de chaque mot du vocabulaire. Le nombre de neurones dans la couche cachée donne la dimension des vecteurs mots finaux. Cet hyper-paramètre est généralement choisi entre 100 et 1000. La dernière couche du réseau utilise la fonction d'activation Softmax, pour calculer la probabilité qu'a un mot c d'apparaître dans le contexte du mot d'entrée w.

$$p(c \mid w) = \frac{exp(v_c'^T v_w)}{\sum_{c_i=1}^{V} exp(v_{c_i}'^T v_w)} \quad (1)$$

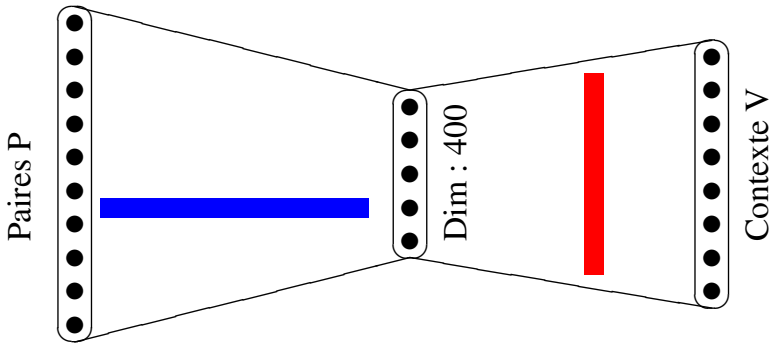

FIGURE 1 – Architecture du réseau utilisé pour générer des vecteurs paires

Où v_w et $v_w^{'}$ sont respectivement les représentations vectorielles de w en tant que mot cible et en tant que contexte.

Enfin, la rétro-propagation s'effectue classiquement en utilisant la fonction de coût de log-vraisemblance négative.

Néanmoins, on peut noter que le coût en calcul de l'équation 1 est très important, à cause de la somme au dénominateur, qui présente un terme pour chaque mot du vocabulaire. Étant donné que la taille du vocabulaire est généralement conséquente, une autre technique proposée par Mikolov *et al.* (2013b), appelée *negative sampling* est nécessaire pour entraîner un modèle avec des coûts de calculs plus raisonnables. Avec le *negative sampling*, le modèle ne doit plus calculer la probabilité $p(c \mid w)$, mais doit prédire si un mot peut apparaître dans le contexte du mot d'entrée. Pour entraîner le modèle, on lui présente donc des exemples "positifs" : des mots qui apparaissent bien dans le même contexte, et des exemples "négatifs" : des mots n'apparaissant pas dans le même contexte. On se retrouve avec un problème de régression logistique pour chaque mot du vocabulaire, ce qui est moins coûteux en terme de calcul.

3.2 Skip-gram pour les paires de mots

Notre but est de générer des représentations vectorielles de relations entre une paire de mots. Nous utilisons une adaptation du modèle Skip-gram classique, en l'appliquant à des paires de mots en entrée du réseau, à la place de mots seuls. Dans ce nouveau modèle, il faut redéfinir l'entrée du réseau, ainsi que la notion de contexte d'une paire de mots. Le modèle que nous utilisons a été pour la première fois implémenté par Chingacham & Paperno (2018) et appliqué aux paires de noms, tandis que nos travaux se concentrent uniquement sur les paires de verbes. En effet nous souhaitions utiliser ce modèle pour diverses tâches de transfert et estimer la pertinence des verbes pour ces tâches. Notre intérêt pour les verbes est dû au fait qu'une partie importante de l'information relationnelle entre deux syntagmes est portée par le verbe. Par exemple, dans la phrase "*Jean **est tombé**, Marie **l'a poussé***.", la relation discursive (causale) entre les deux groupes verbaux est portée par la paire de verbe.

Dans notre modèle, l'entrée du réseau est un encodage *one-hot* d'une paire de mots. Nous avons choisi de ne considérer que le contexte apparaissant entre les deux mots de la paire dans un texte, afin de limiter la taille du vocabulaire du contexte. Néanmoins, de meilleurs résultats pourraient peut-être être obtenus en considérant également les mots apparaissant dans une fenêtre donnée autour de chaque verbe .

Exemple 2 *Sensible guidelines will need to be established for holding case details at a regional level.*

Dans cet exemple, la paire (*will*, *established*) a pour contexte les mots en bleu. Il est important de noter que pour ce modèle, les mots adjacents ne forment pas une paire valide, car ils n'ont pas de contexte, comme la paire (*will*, *need*) dans notre exemple. À part ces redéfinitions, le modèle est exactement le même que le Skip-gram classique avec *negative sampling*.

Avec ce modèle, la fonction de coût est la suivante, où k est le nombre d'exemples négatifs, tirés suivant la distribution P_n :

$$\log \sigma(v_c^{'T} \cdot v_{paire}) + \sum_{i=1}^{k} \mathbb{E}_{c_i \sim P_n} \left(\log \sigma(-v_{c_i}^{'T} \cdot v_{paire}) \right) \tag{2}$$

3.3 Implémentation et construction des vecteurs paires

Le modèle complet décrit dans cet article se découpe en une série d'opérations, composé du traitement d'un important corpus de textes, de l'extraction des paires de verbes, de l'entraînement des représentations vectorielles de paires, puis de l'entraînement d'une matrice de mapping.

3.3.1 Données et pré-traitements

Nous avons utilisé le corpus de texte anglais Ukwac (Ferraresi *et al.*, 2013), contenant 2 milliards de mots. Ce corpus se compose de contenu textuel de pages aspirées du web, et chaque mot est présenté avec sa catégorie grammaticale, ce qui facilite la recherche de verbes.

Avant d'extraire les paires de verbe, nous avons pré-traité le corpus pour enlever certaines informations inutiles pour notre tâche. À l'aide d'un script Python, nous avons transformé le corpus en un document présentant une phrase par ligne où chaque mot est suivi par son étiquette morpho-syntaxique.

Exemple 3 *Portsmouth NP are VBP a DT reminder NN of IN how WRB football NN used VVS to TO be VB.*

Ensuite, pour chaque phrase, les verbes sont identifiés à l'aide de leur étiquette morpho-syntaxique, et les paires de verbes ainsi que les mots de contexte sont comptés. Par ailleurs, afin de réduire la taille du vocabulaire et ne conserver que les paires significatives, seules les paires apparaissant plus de 100 fois sur l'ensemble du corpus sont conservées. De cette manière, le compte final de paires de verbes valides atteint 150 000, et on compte 360 millions de triplets $(verbe_1, verbe_2, contexte)$

Enfin, les exemples d'entraînement positifs et négatifs sont générés pour le *negative sampling*. Nous avons choisi de générer 5 exemples négatifs pour chaque exemple positif.

3.3.2 Entraînement des vecteurs paires

Afin d'entraîner les représentations vectorielles de relations entre deux verbes, nous avons utilisé une adaptation du modèle Skip-gram, détaillée dans la section précédente. Après avoir testé différentes

dimensions de vecteur, comprises entre 50 et 1000, nous avons retenu une dimension de 400, afin d'avoir un bon compromis entre l'information capturée et les temps de calcul. Le modèle a été implémenté à l'aide de la librairie Pytorch. Il implémente une méthode de régularisation ainsi qu'une baisse graduelle du pas d'apprentissage, afin de limiter le sur-apprentissage et d'améliorer la convergence. Cet entraînement a permis de construire un vecteur pour toutes les paires de verbes apparaissant suffisamment souvent dans le corpus. Nous avons entraîné une matrice de mapping afin de construire un vecteur pour toutes les paires de verbes possibles, et ainsi améliorer la couverture des corpus d'évaluation.

3.3.3 Entraînement d'une matrice de mapping

La matrice de mapping est un réseau de neurones prenant en entrée la concaténation de deux vecteurs mots pré-entraînés, et donnant en sortie le vecteur relation entre les deux mots en entrée. L'objectif est de pouvoir obtenir un vecteur relation pour n'importe quelle paire de verbes pour lesquels nous disposons d'un vecteur mot pré-entraîné. Pour les entrées du neurones, nous avons utilisé des vecteurs mots pré-entraînés issus de Baroni *et al.* (2014).

La paire de vecteurs est passée dans un perceptron multi-couches, dont la couche de sortie est la représentation vectorielle de la relation entre les mots d'entrée. Nous avons déterminé empiriquement des valeurs pour les hyper-paramètres et l'architecture du réseau en faisant une évaluation du modèle sur un jeu de données de validation. Finalement nous avons retenu un réseau à une couche cachée, et la fonction d'activation tanh. Nous avons utilisé un pas d'apprentissage de 0.05 et une constante de L2-régularisation de 0.001. Avec cette matrice de mapping, nous pouvons construire un vecteur relation pour toute paire de vecteurs ayant un vecteur mot pré-entraîné.

4 Expériences

4.1 Données

Dans cette section, nous présenterons les corpus utilisés pour l'évaluation des représentations vectorielles de relations entre verbes. Nous présenterons également le corpus utilisé pour la tâche de prédiction de relations discursives.

Pour évaluer la qualité des vecteurs construits avec notre modèle, nous avons utilisé des corpus présentant des paires de mots et la relation correspondante. Nous avions besoin de corpus possédant suffisamment de paires verbe-verbe, donc nous avons utilisé ConceptNet (Speer & Havasi, 2012), un graphe sémantique construit à partir de sources telles que Wikipedia, regroupant des mots dans plus de 300 langues différentes. Dans ce graphe, chaque noeud représente un mot, et chaque arrête représente une relation lexicale. Nous avons extrait de ce corpus uniquement les relations entre deux verbes, ce qui a donné 11 types de relations différents. Nous n'avons pas considéré les paires de verbes ne présentant pas de relations. Nous avons également utilisé le corpus EACL (Nguyen *et al.*, 2017), qui contient des paires de synonymes et d'antonymes, voir Table 1.

Nous avons également utilisé la Penn Discourse Treebank (PDTB) (Prasad *et al.*, 2008), pour la tâche de prédiction de relations du discours. Il s'agit de textes découpés en unités de discours, appelées "arguments", liées par des relations. Notre objectif est d'extraire les verbes de chaque argument et de

Ressource	Type de la relation	Exemples
ConceptNet5	RelatedTo	(eat, feed)
ConceptNet5	Synonym	(jump, leap)
ConceptNet5	MannerOf	(auction, sale)
ConceptNet5	FormOf	(slept, sleep)
ConceptNet5	Antonym	(run, walk)
ConceptNet5	SimilarTo	(compete, win)
ConceptNet5	HasContext	(leave, heading)
ConceptNet5	DistinctFrom	(accept, reject)
ConceptNet5	Entails	(run, move)
ConceptNet5	Causes	(exercise, sweat)
ConceptNet5	DerivedFrom	(paying, pay)
EACL	Synonym	(resurrect, revive)
EACL	Antonym	(love, hate)

TABLE 1 – Jeu de relations et exemples issus de la ressource ConceptNet et du corpus EACL, utilisés pour évaluer nos vecteurs relations

prédire la relation entre les arguments à partir des différentes paires de verbes extraites. Le corpus présente 4 relations de niveau 1, et 16 relations de niveau 2. Ces relations peuvent être implicites en l'absence d'un connecteur discursif, et explicite si les arguments de la relation sont liés par un connecteur.

Le connecteur discursif (*but*, *while*, *because*...) étant porteur de beaucoup d'informations sur le type de relation, nous avons décidé de l'ajouter à notre modèle de vecteur relation. Pour représenter les connecteurs, nous avons assigné à chaque connecteur de la PDTB un index. De cette manière, nous avons pu représenter les connecteurs par des vecteurs *one-hot* de dimension 224, taille du vocabulaire des connecteurs. Par exemple, le connecteur *if* a pour index 85, sa représentation est donc un vecteur de 0, et un 1 à la 85ème position.

Classe (Niveau 1)	Type (Niveau 2)
Comparison	Concession Contrast Pragmatic Concession Pragmatic Contrast
Contingency	Cause Condition Pragmatic Cause Pragmatic Condition
Expansion	Alternative Conjunction Exception Instantiation List Restatement
Temporal	Asynchronous Synchrony

TABLE 2 – Jeu de relations pour les niveaux 1 et 2 dans la PDTB

4.2 Classification de relations

4.2.1 Modèle

Pour évaluer la qualité de notre modèle, nous avons donc utilisé un réseau de neurones réalisant une classification multi-classes prenant en entrée la représentation vectorielle d'une paire de verbes, et essayant de prédire la bonne relation. Les vecteurs en entrée on été construits grâce à la matrice de mapping détaillée en Section 3.3.3.

Nous utilisons un perceptron multi-couches avec une dernière couche Softmax pour réaliser la classification. Classiquement, nous utilisons la fonction de coût de log-vraisemblance à minimiser.

$$-\sum_{y \in train} y \log y_{prediction} \tag{3}$$

Pour le jeu de données PDTB, nous avons tokenisé les phrases avec la librairie Python NLTK, afin de pouvoir trouver et extraire les verbes. Il arrive que des arguments d'une relation contiennent plusieurs verbes. Dans ce cas, nous prenons le vecteur moyen de toutes les paires de verbes entre les arguments. Dans le cas où une relation ne peut pas être représentée par la matrice de mapping, en l'absence des vecteurs pré-entraînés nécessaires, alors nous testons si la paire lemmatisée peut avoir un vecteur par la matrice de mapping. Si ce n'est pas le cas, alors nous donnons à la relation un vecteur moyen, calculé en faisant la moyenne de tout les vecteurs construits par le modèle.

4.2.2 Hyper-paramètres

Pour entraîner le classifieur, nous avons utilisé la descente du gradient stochastique avec une taille de mini-batch de 8, ainsi qu'une régularisation avec norme L2 de constante 0.02, déterminée sur un jeu de données de validation. Par ailleurs, nous avons utilisé la répartition 60% de données à l'entraînement, 10% pour l'ensemble de validation et 30% pour l'ensemble de test. Le classifieur est entraîné sur 100 itérations, avec un pas d'apprentissage décroissant, fixé à 0.01 au départ. Plusieurs architectures de réseaux ont été testées pour la classification, et nous avons finalement gardé un réseau à une couche cachée, de même taille que la couche d'entrée, avec la fonction d'activation ReLU.

Nous avons également testé plusieurs dimensions pour les vecteurs relations, mais peu de différences ont été observées sur le jeu de données de développement. Nous avons finalement retenu une dimension 400, pour pouvoir se comparer au modèle utilisant uniquement la différence de deux vecteurs mots, car les vecteurs mots pré-entraînés dont nous disposions sont de dimension 400.

Dimension du vecteur	50	100	200	300	400	1000
Précision sur l'ensemble de validation	55.08	55.12	56.51	56.70	56.96	57.02

4.2.3 Lemmatisation des verbes

Une intuition au sujet des paires de verbes a été qu'il serait possible d'obtenir de meilleurs résultats en ne construisant que des vecteurs de paires de verbes lemmatisés, c'est à dire dans leur forme neutre. Ce choix a été motivé par le fait que les corpus utilisés présentent en majorité des paires de verbes

sous cette forme. Nous comparerons dans la section résultats les scores obtenus avec des verbes lemmatisés et non lemmatisés.

4.2.4 Exécution

Les unités de calcul nécessaires aux expériences présentées dans cet article, ainsi que l'espace de stockage pour les jeux de données, ont été fournis par Grid5000 (Balouek *et al.*, 2013). L'implémentation est réalisée sur Pytorch, et le code est vectorisé le plus possible pour diminuer les temps d'exécution sur des GPUs.

5 Résultats

Dans cette section seront présentés les résultats de l'évaluation des représentations vectorielles de relations entre deux verbes sur les jeux de données ConceptNet et EACL. Nous donnerons également les résultats obtenus sur la tâche de prédiction des relations du discours sur la *Penn Discourse Treebank*.

5.1 Systèmes de référence

Pour évaluer la qualité de nos résultats, nous les comparerons, pour chaque jeu de données, au deux systèmes de référence suivants :

— La *random baseline*, score obtenu par un modèle prédisant au hasard.
— La *majority baseline*, score obtenu par un modèle prédisant toujours la classe majoritaire du jeu de données.

Les scores présentés sont les scores F1 obtenus sur les données de test. Nous comparerons trois modèles différents de représentation de relations :

— **SkipRel** : Le modèle entraîné par notre variante de Skip-gram pour les paires de verbes. Les vecteurs sont de dimension 400.
— **JustWord** : La différence des deux vecteurs mots de la paire de verbes. Ce modèle utilise des *word embeddings* pré-entraînés de Baroni *et al.* (2014), de dimension 400.
— **RelWord** : La concaténation des deux précédents modèles, résultant en un vecteur de dimension 800.

5.2 Évaluation sur ConceptNet et EACL

Les résultats détaillés de l'évaluation des vecteurs sur la ressource ConceptNet et le corpus EACL sont donnés dans le tableau 3. Nous avons observé que la couverture du jeu de données EACL était mauvaise (moins de 50% des paires de verbes avaient une représentation vectorielle), donc les expériences sur ce jeu de données ont été menées en utilisant la matrice de mapping entraînée pour améliorer la couverture. Par ailleurs, nous avons testé l'évaluation sur les données de ConceptNet avec et sans la matrice de mapping, afin d'observer le gain en performance apporté par les vecteurs construits par la matrice.

Comme expliqué dans la section 4.2.3, nous avons eu l'intuition que les résultats pourraient être améliorés sur la ressource ConceptNet et le corpus EACL en lemmatisant les paires de verbes. Pour cela nous avons lemmatisé les verbes extraits du corpus Ukwac à l'aide de la librairie Python NLTK. Les résultats de l'évaluation avec le modèle entraînés sur des paires de verbes lemmatisés sont également fournis ci-dessous.

Ressource	ConceptNet			EACL	
Modèle	Sans mapping	Mapping	Lemmatisé	Mapping	Lemmatisé
JustWord	50.54	56.46	63.60	**87.32**	**86.85**
SkipRel	48.08	55.70	55.53	70.42	74.36
RelWord	**52.75**	**60.79**	**66.60**	86.88	85.97
RandomBaseline	9.09	9.09	9.09	50	50
MajorityBaseline	34.73	25.3	25.3	50.86	50.86

TABLE 3 – Résultats de l'évaluation des vecteurs paires de verbes

Premièrement, on voit sur ConceptNet que les modèles JustWord et SkipRel obtiennent des scores proches. Néanmoins on observe que le modèle RelWord obtient les meilleurs résultats. Cela signifie que les vecteurs relations que nous avons construits contiennent de l'information différente de l'information contenue par les vecteurs mots individuels. Deuxièmement, les résultats sont améliorés par l'utilisation de la matrice de mapping, car la couverture des données est meilleure. L'utilisation de la lemmatisation sur ConceptNet améliore de façon importante les résultats du modèle JustWord, mais ne semble pas avoir d'impact sur les résultats du modèle SkipRel. Bien que le modèle JustWord obtienne systématiquement de meilleurs scores que notre modèle SkipRel, la concaténation des deux modèles donne les meilleures performances sur ConceptNet, ce qui rend le modèle SkipRel intéressant. Enfin, notre modèle SkipRel donne des résultats très inférieurs à ceux du modèle JustWord sur la tâche de classification d'antonymes et de synonymes, bien que sur cette tâche la lemmatisation des verbes permet une amélioration du score.

En observant l'évolution de la fonction de coût lors de l'entraînement du classifieur sur les données EACL, on se rend compte que le modèle SkipRel fait du sur-apprentissage, ce qui impacte négativement les résultats.

5.3 Prédiction des relations du discours sur la Penn Discourse Treebank

Les scores atteints sur la tâche de prédiction de relations du discours sur les données de la *Penn Discourse Treebank* sont donnés dans le tableau 4.

On distingue les résultats obtenus par les modèles JustWord, SkipRel et RelWord. Dans le cas de relations explicites, c'est à dire les relations qui présentent un connecteur discursif, une représentation *one-hot* de dimension 224 de ce connecteur est concaténée au vecteur SkipRel.

L'évaluation est faite séparément sur les relations implicites et explicites, et nous avons par ailleurs réalisé séparément l'évaluation sur les relations de niveau 2, qui sont plus précises que les relations de niveau 1.

Nous comparons également nos résultats avec les résultats obtenus lors de la *Conll-Shared Task 2016* (Xue *et al.*, 2016).

Type de relation	Implicite		Explicite	
Modèle	Level 1	Level 2	Level 1	Level 2
JustWord	59.71	36.56	62.97	49.81
SkipRel + connecteur (si rel. explicite)	59.01	38.16	74.12	54.01
RelWord	**61.36**	**40.20**	**79.59**	**61.32**
RandomBaseline	25.0	6.67	25.0	6.67
MajorityBaseline	34.73	25.3	35.1	27.3
ConLL Shared Task 2016	-	40.80 [1]	-	78.56

TABLE 4 – Résultats de la tâche de prédiction de relations du discours

On peut sur ce tableau observer l'apport important d'information venant du connecteur, puisque le modèle SkipRel obtient de bien meilleurs résultats sur la prédiction de relations explicites. À nouveau les vecteurs relations que nous avons entraînés semblent être complémentaires avec les *word embeddings* individuels de la paire, car RelWord obtient les meilleurs résultats sur cette tâche, quelque soit le type de relation. Au sujet de la répartition des types de relation sur l'ensemble de test, les relations explicites sont bien équilibrées, les relations de type *Comparison* étant les mieux représentées avec 75% des relations correctement classées dans cette catégorie. Pour les relations implicites, les relations *Temporal* sont sous-représentées, avec moins de 50 exemples, contre plus de 500 relations de type *Expansion*. Malgré tout, 20% des relations *Temporal* sont correctement classées, bien que la relation de type *Comparison* est à nouveau la mieux représentée avec 70% des relations correctement prédites. Nous sommes cependant encore loin des performances état de l'art sur la tâche (voir dernière ligne de 4), ce qui s'explique par le fait que nous ne prenons en compte qu'une partie de l'information disponible dans les arguments. Les scores sont cependant significativement au-dessus de la chance, démontrant que les représentation encodent des informations pertinentes.

Afin de visualiser la qualité de la concaténation des vecteurs relations et du vecteur de connecteur discursif, nous avons représenté ces vecteurs, de dimension 624, en 3D à l'aide d'une analyse en composantes principales (ACP), voir Figure 2. Les couleurs sont les labels des relations explicites de niveau 1. On observe que des groupes se forment, alors qu'une ACP uniquement sur les vecteurs connecteurs ou sur les vecteurs relations ne permet pas de distinguer de clusters. Les relations de type *Comparison* et *Expansion* se distinguent clairement, et les relations de type *Temporal* et *Contingency* se mélangent. Cela fait sens sémantiquement, car le type *Contingency* correspond aux relations causales, or on retrouve souvent une dimension temporelle dans les relations causales.

D'autre part, nous avons réalisé une réduction de dimension avec l'algorithme t-SNE sur les mêmes vecteurs, cette fois pour les relations de niveau 2 (fig. 3). Les couleurs représentent les labels des relations. À nouveau, des clusters se dessinent entre les types de relations les plus représentés, ce qui tend à démontrer que les représentations relationnelles construites sur les paires de verbes encodent une information pertinente pour la dimension rhétorique.

1. Score obtenu sur un jeu plus large de types de relations

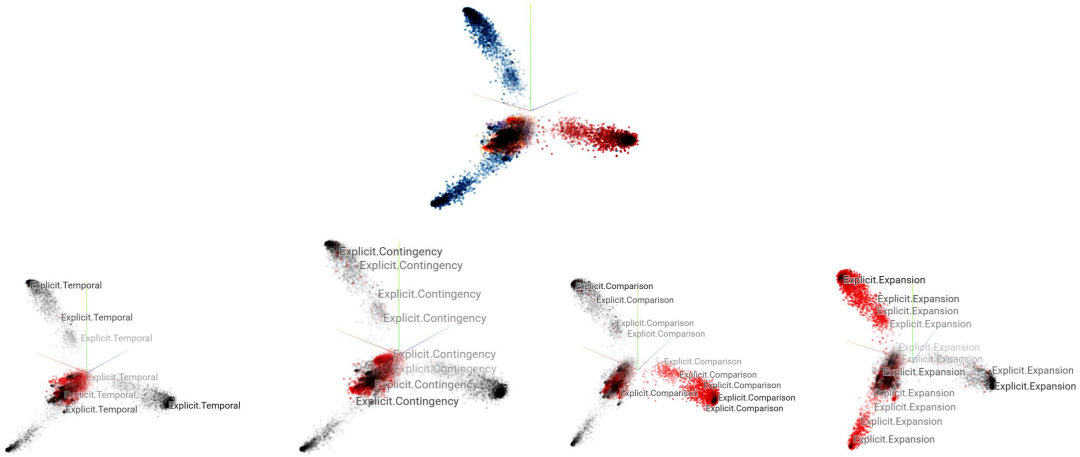

FIGURE 2 – ACP sur les vecteurs SkipRel + Connecteur pour les relations explicites de niveau 1

FIGURE 3 – Résultat de l'algorithme t-SNE sur les vecteurs SkipRel + Connecteur pour les relations explicites de niveau 2

6 Conclusion

Dans cet article, nous avons proposé une approche visant à construire une représentation distribuée, non pour des mots, mais pour des paires de mots dans le but d'encoder leur relation. Nous avons évalué ces représentations sur 2 tâches venant de 3 corpus : la prédiction de relations lexicales (antonymies, synonymies) et la classification de relations discursives (temporelles, causales...). Le modèle utilisé pour la construction de ces vecteurs relations est une variante de Skip-gram, et nos recherches se concentrent sur les paires de verbes. Les résultats montrent que ces vecteurs relations capturent une information différente des vecteurs mots classiques, ce qui rend la composition des deux représentations plus performante sur diverses tâches de transfert. L'évaluation sur la tâche de prédiction des relations du discours a révélé des résultats intéressants sur la façon dont les vecteurs de verbes et les connecteurs logiques se complètent. La visualisation à l'aide d'une analyse en composantes principales en 3 dimensions appuie cette observation. Néanmoins, notre modèle ne permet pas d'approcher les résultats obtenus par des modèles plus complexes, car l'information contenue dans les verbes d'une phrase n'est souvent pas suffisante pour en déduire l'articulation. Des modèles plus avancés capables de générer des représentations vectorielles de phrases entières pourraient compléter notre modèle en permettant la prise en compte de tous les mots de la phrase. La concaténation de ces deux modèles pourrait atteindre de bonnes performances sur la tâche de prédiction de relations du discours.

Remerciements

Je remercie Denis Paperno et Chloé Braud qui ont encadré ce travail, pour leur disponibilité et les précieux conseils qu'ils m'ont fournis.

Nous remercions les relecteurs pour leurs commentaires pertinents.

Références

BALOUEK D., CARPEN AMARIE A., CHARRIER G., DESPREZ F., JEANNOT E., JEANVOINE E., LÈBRE A., MARGERY D., NICLAUSSE N., NUSSBAUM L., RICHARD O., PÉREZ C., QUESNEL F., ROHR C. & SARZYNIEC L. (2013). Adding virtualization capabilities to the Grid'5000 testbed. In I. I. IVANOV, M. VAN SINDEREN, F. LEYMANN & T. SHAN, Éds., *Cloud Computing and Services Science*, volume 367 de *Communications in Computer and Information Science*, p. 3–20. Springer International Publishing. DOI : 10.1007/978-3-319-04519-1_1.

BARONI M., DINU G. & KRUSZEWSKI G. (2014). Don't count, predict ! a systematic comparison of context-counting vs. context-predicting semantic vectors. volume 1, p. 238–247. DOI : 10.3115/v1/P14-1023.

BARONI M. & ZAMPARELLI R. (2010). Nouns are vectors, adjectives are matrices : Representing adjective-noun constructions in semantic space. p. 1183–1193.

BENGIO Y., DUCHARME R., VINCENT P. & JAUVIN C. (2006). *Neural Probabilistic Language Models*, volume 3, p. 137–186. DOI : 10.1007/3-540-33486-6_6.

BRAUD C. & DENIS P. (2016). Learning connective-based word representations for implicit discourse relation identification.

BULLINARIA J. & LEVY J. (2012). Extracting semantic representations from word co-occurrence statistics : Stop-lists, stemming, and svd. *Behavior research methods*, **44**, 890–907. DOI : 10.3758/s13428-011-0183-8.

CHINGACHAM A. & PAPERNO D. (2018). Generalizing representations of lexical semantic relations.

CONNEAU A., KIELA D., SCHWENK H., BARRAULT L. & BORDES A. (2017). Supervised learning of universal sentence representations from natural language inference data.

FERRARESI A., ZANCHETTA E., BARONI M. & BERNARDINI S. (2013). Introducing and evaluating ukwac, a very large web-derived corpus of english.

HARRIS Z. S. (1954). Distributional structure. *Word*, **10(23)**, 146–162.

HILL F., CHO K. & KORHONEN A. (2016). Learning distributed representations of sentences from unlabelled data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, p. 1367–1377, San Diego, California : Association for Computational Linguistics. DOI : 10.18653/v1/N16-1162.

JAMEEL S., BOURAOUI Z. & SCHOCKAERT S. (2018). Unsupervised learning of distributional relation vectors.

JOSHI M., CHOI E. & LEVY O. (2018). pair2vec : Compositional word-pair embeddings for cross-sentence inference.

LENCI A. (2018). Distributional models of word meaning. *Annual Review of Linguistics*, **4**. DOI : 10.1146/annurev-linguistics-030514-125254.

MIKOLOV T., CHEN K., CORRADO G. & DEAN J. (2013a). Efficient estimation of word representations in vector space.

MIKOLOV T., SUTSKEVER I., CHEN K., CORRADO G. & DEAN J. (2013b). Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, **26**.

NGUYEN K. A., SCHULTE IM WALDE S. & VU T. (2017). Distinguishing antonyms and synonyms in a pattern-based neural network. DOI : 10.18653/v1/E17-1008.

NIE A., BENNETT E. & GOODMAN N. (2019). Dissent : Learning sentence representations from explicit discourse relations.

PAPERNO D., PHAM N. & BARONI M. (2014). A practical and linguistically-motivated approach to compositional distributional semantics. volume 1, p. 90–99. DOI : 10.3115/v1/P14-1009.

PENNINGTON J., SOCHER R. & MANNING C. (2014). Glove : Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, p. 1532–1543, Doha, Qatar : Association for Computational Linguistics. DOI : 10.3115/v1/D14-1162.

PRASAD R., DINESH N., LEE A., MILTSAKAKI E., ROBALDO L., JOSHI A. & WEBBER B. (2008). The penn discourse treebank 2.0.

RITTER S., LONG C., PAPERNO D., BARONI M., BOTVINICK M. & GOLDBERG A. (2015). Leveraging preposition ambiguity to assess compositional distributional models of semantics. p. 199–204. DOI : 10.18653/v1/S15-1023.

SILEO D., VAN DE CRUYS T., PRADEL C. & MULLER P. (2019). Mining discourse markersfor unsupervised sentence representation learning.

SPEER R. & HAVASI C. (2012). Representing general relational knowledge in conceptnet 5. *Proc. of LREC*, p. 3679–3686.

TURNEY P. D. & PANTEL P. (2010). From frequency to meaning : Vector space models of semantics. *Journal of Artificial Intelligence Research*, p. 141–188.

WESTON J., BORDES A., YAKHNENKO O. & USUNIER N. (2013). Connecting language and knowledge bases with embedding models for relation extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, p. 1366–1371, Seattle, Washington, USA : Association for Computational Linguistics.

XUE N., NG H., PRADHAN S., RUTHERFORD A., WEBBER B., WANG C. & WANG H. (2016). Conll 2016 shared task on multilingual shallow discourse parsing. p. 1–19. DOI : 10.18653/v1/K16-2001.