Distributed Representations of Words and Phrases and their Compositionality

Abstract

Le modèle Skip-gram continu récemment introduit est une méthode efficace pour apprendre des représentations vectorielles distribuées de haute qualité qui capturent un grand nombre de relations syntaxiques et sémantiques précises entre les mots. Dans cet article, nous présentons plusieurs extensions qui améliorent à la fois la qualité des vecteurs et la vitesse d'apprentissage. En sous-échantillonnant les mots fréquents, nous obtenons une accélération significative et apprenons également des représentations de mots plus régulières. Nous décrivons également une alternative simple au softmax hiérarchique appelée échantillonnage négatif.

Une limitation inhérente aux représentations de mots est leur indifférence à l'ordre des mots et leur incapacité à représenter des phrases idiomatiques. Par exemple, les significations de « Canada » et « Air » ne peuvent pas être facilement combinées pour obtenir « Air Canada ». Motivés par cet exemple, nous présentons une méthode simple pour trouver des phrases dans un texte et montrons qu'il est possible d'apprendre de bonnes représentations vectorielles pour des millions de phrases.

1 Introduction

Les représentations distribuées des mots dans un espace vectoriel aident les algorithmes d'apprentissage à obtenir de meilleures performances dans les tâches de traitement du langage naturel en regroupant les mots similaires. L'une des premières utilisations des représentations de mots remonte à 1986, grâce à Rumelhart, Hinton et Williams [13]. Cette idée a depuis été appliquée à la modélisation statistique du langage avec un succès considérable [1]. Les travaux qui ont suivi comprennent des applications à la reconnaissance automatique de la parole et à la traduction automatique [14, 7], ainsi qu'à un large éventail de tâches de traitement automatique des langues [2, 20, 15, 3, 18, 19, 9].

Récemment, Mikolov et al. [8] ont présenté le modèle Skip-gram, une méthode efficace pour apprendre des représentations vectorielles de haute qualité des mots à partir de grandes quantités de données textuelles non structurées. Contrairement à la plupart des architectures de réseaux neuronaux utilisées précédemment pour l'apprentissage des vecteurs de mots, l'apprentissage du modèle Skip-gram (voir figure 1) n'implique pas de

multiplications matricielles denses. Cela rend l'apprentissage extrêmement efficace : une implémentation optimisée sur une seule machine peut s'entraîner sur plus de 100 milliards de mots en une journée.

Les représentations de mots calculées à l'aide de réseaux neuronaux sont très intéressantes car les vecteurs appris codent explicitement de nombreuses régularités et modèles linguistiques. De manière assez surprenante, nombre de ces modèles peuvent être représentés par des traductions linéaires. Par exemple, le résultat d'un calcul vectoriel vec("Madrid") - vec("Espagne") + vec("France") est plus proche de vec("Paris") que de tout autre vecteur de mots [9, 8].

=====

Figure 1 : L'architecture du modèle Skip-gram. L'objectif de la formation est d'apprendre des représentations de vecteurs de mots qui sont bonnes pour prédire les mots proches.

Dans cet article, nous présentons plusieurs extensions du modèle original Skip-gram. Nous montrons que le sous-échantillonnage des mots fréquents pendant l'apprentissage entraîne une accélération significative (environ 2x - 10x), et améliore la précision des représentations des mots moins fréquents. En outre, nous présentons une variante simplifiée de l'Estimation Contrastive du Bruit (NCE) [4] pour l'entraînement du modèle Skip-gram qui permet un entraînement plus rapide et de meilleures représentations vectorielles pour les mots fréquents, par rapport à la méthode hiérarchique plus complexe de softmax utilisée dans les travaux précédents [8].

Les représentations de mots sont limitées par leur incapacité à représenter des expressions idiomatiques qui ne sont pas des combinaisons de mots individuels. Par exemple, "Boston Globe" est un journal, et n'est donc pas une combinaison naturelle des significations de "Boston" et "Globe". Par conséquent, l'utilisation de vecteurs pour représenter les phrases entières rend le modèle Skip-gram considérablement plus expressif. D'autres techniques qui visent à représenter le sens des phrases en composant les vecteurs de mots, comme les autoencodeurs récursifs [15], bénéficieraient également de l'utilisation de vecteurs de phrases au lieu de vecteurs de mots.

L'extension des modèles basés sur les mots aux modèles basés sur les phrases est relativement simple. Tout d'abord, nous identifions un grand nombre de phrases en utilisant une approche basée sur les données, puis nous traitons les phrases comme des

tokens individuels pendant l'apprentissage. Pour évaluer la qualité des vecteurs de phrases, nous avons développé un ensemble de test de tâches de raisonnement analogique qui contient à la fois des mots et des phrases. Une paire d'analogies typique de notre ensemble de test est "Montréal" : "Canadiens de Montréal": : "Toronto" : "Maple Leafs de Toronto". On considère que la réponse est correcte si la représentation la plus proche de vec("Montreal Canadiens") - vec("Montreal") + vec("Toronto") est vec("Toronto Maple Leafs").

Enfin, nous décrivons une autre propriété intéressante du modèle Skip-gram. Nous avons constaté que la simple addition de vecteurs peut souvent produire des résultats significatifs. Par exemple, vec("Russia") + vec("river") est proche de vec("Volga River"), et vec("Germany") + vec("capital") est proche de vec("Berlin"). Cette composition suggère qu'un degré non évident de compréhension de la langue peut être obtenu en utilisant des opérations mathématiques de base sur les représentations vectorielles des mots.

2 The Skip-gram Model

L'objectif de formation du modèle Skip-gram est de trouver des représentations de mots qui sont utiles pour prédire les mots environnants dans une phrase ou un document. Plus formellement, étant donné une séquence de mots d'apprentissage w1 , w2 , w3 , . . . , wT , l'objectif du modèle Skip-gram est de maximiser la probabilité logarithmique moyenne

====

où c est la taille du contexte d'apprentissage (qui peut être une fonction du mot central wt). Une taille plus grande de
c plus grande entraîne un plus grand nombre d'exemples d'apprentissage et peut donc conduire à une plus grande précision, au détriment du temps d'apprentissage. La formulation de base du saut de programme définit p(wt+j |wt) en utilisant la fonction softmax :

====

où vw et vw’ sont les représentations vectorielles " entrée " et " sortie " de w, et W est le nombre de mots du vocabulaire. Cette formulation n'est pas pratique car le coût de calcul de ∇logp(wO|wI)estproportionnel à W, qui est souvent grand (105-107 termes).

2.1 Hierarchical Softmax

Une approximation efficace du point de vue informatique de la maxime douce complète est la maxime douce hiérarchique. Dans le contexte des modèles de langage des réseaux de neurones, elle a été introduite pour la première fois par Morin et Bengio [12]. Son principal avantage est qu'au lieu d'évaluer W nœuds de sortie dans le réseau neuronal pour obtenir la distribution de probabilité, il n'est nécessaire d'évaluer qu'environ log2(W) nœuds.

Le softmax hiérarchique utilise une représentation arborescente binaire de la couche de sortie avec les mots W comme feuilles et, pour chaque nœud, représente explicitement les probabilités relatives de ses nœuds enfants. Celles-ci définissent une marche aléatoire qui attribue des probabilités aux mots.

Plus précisément, chaque mot w peut être atteint par un chemin approprié à partir de la racine de l'arbre. Soit n(w, j) le j-ième nœud sur le chemin de la racine à w, et soit L(w) la longueur de ce chemin, donc n(w, 1) = racine et n(w, L(w)) = w. En outre, pour tout nœud interne n, soit ch(n) un enfant fixe arbitraire de n et soit [x] 1 si x est vrai et -1 sinon. Alors, le softmax hiérarchique définit p(wO |wI) comme suit :

====

où σ(x) = 1/(1 + exp(-x)). On peut vérifier que Ww=1 p(w|wI) = 1. Cela implique que le coût du calcul de log p(wO |wI) et de ∇ log p(wO |wI) est proportionnel à L(wO), qui n'est en moyenne pas supérieur à log W . De plus, contrairement à la formulation softmax standard du saut-gramme qui attribue deux représentations vw et vw’ à chaque mot w, la formulation softmax hiérarchique a une représentation vw pour chaque mot w et une représentation vn’ pour chaque nœud interne n de l'arbre binaire.

La structure de l'arbre utilisée par le softmax hiérarchique a un effet considérable sur les performances. Mnih et Hinton ont exploré un certain nombre de méthodes pour construire la structure de l'arbre et l'effet sur le temps de formation et la précision du modèle résultant [10]. Dans notre travail, nous utilisons un arbre de Huffman binaire, car il attribue des codes courts aux mots fréquents, ce qui permet un apprentissage rapide. Il a déjà été observé que le regroupement des mots par leur fréquence fonctionne bien

comme une technique d'accélération très simple pour les modèles de langage basés sur un réseau de neurones [5, 8].

2.2 Negative Sampling

Une alternative à la softmax hiérarchique est l'Estimation Contrastive du Bruit (NCE), qui a été introduite par Gutmann et Hyvarinen [4] et appliquée à la modélisation du langage par Mnih et Teh [11]. La NCE postule qu'un bon modèle doit être capable de différencier les données du bruit au moyen d'une régression logistique. Ceci est similaire à la perte de charnière utilisée par Collobert et Weston [2] qui ont entraîné les modèles en classant les données au-dessus du bruit.

Bien que l'on puisse montrer que NCE maximise approximativement la probabilité logarithmique de la softmax, le modèle Skip- gram ne s'intéresse qu'à l'apprentissage de représentations vectorielles de haute qualité, nous sommes donc libres de simplifier NCE tant que les représentations vectorielles conservent leur qualité. Nous définissons l'échantillonnage négatif (NEG) par l'objectif

====
====

Figure 2 : Projection PCA bidimensionnelle des vecteurs Skip-gram à 1000 dimensions des pays et de leurs capitales. La figure illustre la capacité du modèle à organiser automatiquement les concepts et à apprendre implicitement les relations entre eux, étant donné qu'au cours de la formation, nous n'avons fourni aucune information supervisée sur la signification d'une capitale.

qui est utilisé pour remplacer chaque terme log P (wO |wI) dans l'objectif du saut de programme. La tâche consiste donc à distinguer le mot cible wO des tirages de la distribution du bruit Pn(w) à l'aide d'une régression logistique, où il y a k échantillons négatifs pour chaque échantillon de données. Nos expériences indiquent que des valeurs de k comprises entre 5 et 20 sont utiles pour les petits ensembles de données d'entraînement, tandis que pour les grands ensembles de données, k peut être aussi petit que 2-5. La principale différence entre l'échantillonnage négatif et NCE est que NCE a besoin à la fois d'échantillons et des probabilités numériques de la distribution du bruit, alors que l'échantillonnage négatif n'utilise que des échantillons. Et bien que NCE

maximise approximativement la probabilité logarithmique de la softmax, cette propriété n'est pas importante pour notre application.

NCE et NEG ont tous deux la distribution du bruit Pn(w) comme paramètre libre. Nous avons étudié un certain nombre de choix pour Pn(w) et avons constaté que la distribution unigramme U(w) élevée à la puissance 3/4 (c.-à-d. U(w)3/4/Z) surpasse de manière significative les distributions unigramme et uniforme, à la fois pour NCE et NEG sur chaque tâche que nous avons essayée, y compris la modélisation du langage (non rapportée ici).

2.3 Subsampling of Frequent Words

Dans les très grands corpus, les mots les plus fréquents peuvent facilement apparaître des centaines de millions de fois (par exemple, "in", "the" et "a"). Ces mots fournissent généralement moins de valeur informative que les mots rares. Par exemple, si le modèle Skip-gram bénéficie de l'observation des cooccurrences de "France" et "Paris", il bénéficie beaucoup moins de l'observation des cooccurrences fréquentes de "France" et "le", car presque tous les mots cooccurrent fréquemment dans une phrase avec "le". Cette idée peut également être appliquée dans le sens inverse ; les représentations vectorielles des mots fréquents ne changent pas de manière significative après un entraînement sur plusieurs millions d'exemples.

Pour contrer le déséquilibre entre les mots rares et fréquents, nous avons utilisé une approche de sous-échantillonnage simple : chaque mot wi de l'ensemble de formation est éliminé avec une probabilité calculée par la formule suivante

====
====

Tableau 1 : Précision de divers modèles Skip-gram 300-dimensionnels sur la tâche de raisonnement analogique telle que définie dans [8]. NEG-k signifie Echantillonnage Négatif avec k échantillons négatifs pour chaque échantillon positif ; NCE signifie Estimation Contrastive du Bruit et HS-Huffman signifie Softmax Hiérarchique avec les codes de Huffman basés sur la fréquence.

où f(wi) est la fréquence du mot wi et t est un seuil choisi, généralement autour de 10-5. Nous avons choisi cette formule de sous-échantillonnage car elle sous-échantillonne de

manière agressive les mots dont la fréquence est supérieure à t tout en préservant le classement des fréquences. Bien que cette formule de sous-échantillonnage ait été choisie de manière heuristique, nous avons constaté qu'elle fonctionne bien en pratique. Elle accélère l'apprentissage et améliore même significativement la précision des vecteurs appris des mots rares, comme nous le montrerons dans les sections suivantes.

3 Empirical Results

Dans cette section, nous évaluons les méthodes suivantes : Hierarchical Softmax (HS), Noise Contrastive Estimation, Negative Sampling et subsampling des mots d'entraînement. Nous avons utilisé la tâche de raisonnement analogique1 introduite par Mikolov et al [8]. La tâche consiste en des analogies telles que "Allemagne" : "Berlin" : : "France" : ?, qui sont résolues en trouvant un vecteur x tel que vec(x) est le plus proche de vec("Berlin") - vec("Allemagne") + vec("France") selon la distance en cosinus (nous écartons les mots d'entrée de la recherche). On considère que la réponse à cet exemple spécifique est correcte si x est "Paris". La tâche comporte deux grandes catégories : les analogies syntaxiques (telles que "rapide" : "rapidement" : : "lent" : "lentement") et les analogies sémantiques, telles que la relation pays-capitale.

Pour l'entraînement des modèles Skip-gram, nous avons utilisé un grand ensemble de données composé de divers articles de presse (un ensemble de données interne de Google contenant un milliard de mots). Nous avons éliminé du vocabulaire tous les mots qui apparaissaient moins de 5 fois dans les données d'entraînement, ce qui a donné un vocabulaire de 692 000 mots. Les performances des différents modèles de saut de programme sur l'ensemble de test d'analogie de mots sont présentées dans le tableau 1. Le tableau montre que l'échantillonnage négatif surpasse le modèle hiérarchique Softmax sur la tâche de raisonnement analogique, et a même des performances légèrement meilleures que l'estimation contrastive du bruit. Le sous-échantillonnage des mots fréquents améliore plusieurs fois la vitesse d'apprentissage et rend les représentations des mots beaucoup plus précises.

On peut avancer que la linéarité du modèle de saut de programme rend ses vecteurs plus adaptés à ce type de raisonnement analogique linéaire, mais les résultats de Mikolov et al. [8] montrent également que les vecteurs appris par les réseaux neuronaux récurrents sigmoïdaux standard (qui sont hautement non linéaires) s'améliorent de manière significative dans cette tâche lorsque la quantité de données d'apprentissage augmente,

ce qui suggère que les modèles non linéaires ont également une préférence pour une structure linéaire des représentations des mots.

4 Learning Phrases

Comme nous l'avons vu précédemment, de nombreuses phrases ont une signification qui n'est pas une simple composition des significations de leurs mots individuels. Pour apprendre la représentation vectorielle des phrases, nous trouvons d'abord des mots qui apparaissent fréquemment ensemble, et peu fréquemment dans d'autres contextes. Par exemple, "New York Times" et "Toronto Maple Leafs" sont remplacés par des jetons uniques dans les données d'apprentissage, tandis que le bigramme "ceci est" restera inchangé.

====

Tableau 2 : Exemples de la tâche de raisonnement analogique pour les phrases (le jeu de test complet comporte 3218 exemples). L'objectif est de calculer la quatrième phrase en utilisant les trois premières. Notre meilleur modèle a obtenu une précision de 72% sur ce jeu de données.

De cette façon, nous pouvons former de nombreuses phrases raisonnables sans augmenter considérablement la taille du vocabulaire. En théorie, nous pouvons entraîner le modèle Skip-gram en utilisant tous les n-grammes, mais cela demanderait trop de mémoire. De nombreuses techniques ont déjà été développées pour identifier les phrases dans le texte, mais il n'entre pas dans le cadre de notre travail de les comparer. Nous avons décidé d'utiliser une approche simple basée sur les données, dans laquelle les phrases sont formées sur la base du nombre d'unigrammes et de bigrammes, en utilisant le modèle

====

Le δ est utilisé comme un coefficient d'actualisation et empêche la formation d'un trop grand nombre de phrases composées de mots très infréquentables. Les bigrammes dont le score est supérieur au seuil choisi sont alors utilisés comme phrases. Typiquement, nous effectuons 2 à 4 passages sur les données d'entraînement avec une valeur de seuil décroissante, permettant la formation de phrases plus longues composées de plusieurs

mots. Nous évaluons la qualité des représentations de phrases en utilisant une nouvelle tâche de raisonnement analogique qui implique des phrases. Le tableau 2 montre des exemples des cinq catégories d'analogies utilisées dans cette tâche. Cet ensemble de données est disponible publiquement sur le web2.

4.1 Phrase Skip-Gram Results

En partant des mêmes données d'actualités que dans les expériences précédentes, nous avons d'abord construit le corpus d'entraînement basé sur les phrases, puis nous avons entraîné plusieurs modèles Skip-gram en utilisant différents hyper-paramètres. Comme précédemment, nous avons utilisé une dimension de vecteur de 300 et une taille de contexte de 5. Ce paramètre permet déjà d'obtenir de bonnes performances sur le jeu de données de phrases, et nous a permis de comparer rapidement l'échantillonnage négatif et le Softmax hiérarchique, avec et sans sous-échantillonnage des tokens fréquents. Les résultats sont résumés dans le tableau 3.

Les résultats montrent que si l'échantillonnage négatif permet d'obtenir une précision respectable même avec k = 5, l'utilisation de k = 15 permet d'obtenir des performances considérablement meilleures. De manière surprenante, alors que la méthode Hierar- chical Softmax est moins performante lorsqu'elle est entraînée sans sous-échantillonnage, elle devient la méthode la plus performante lorsque nous sous-échantillonnons les mots fréquents. Cela montre que le sous-échantillonnage peut permettre une formation plus rapide et améliorer la précision, du moins dans certains cas.

====

Tableau 3 : Précision des modèles Skip-gram sur l'ensemble de données d'analogie de phrases. Les modèles ont été entraînés sur environ un milliard de mots de l'ensemble de données d'actualités.

====

Tableau 4 : Exemples d'entités les plus proches des phrases courtes données, en utilisant deux modèles différents.

====

Tableau 5 : Composition des vecteurs en utilisant l'addition par éléments. Les quatre tokens les plus proches de la somme de deux vecteurs sont indiqués, en utilisant le meilleur modèle Skip-gram.

Pour maximiser la précision de la tâche d'analogie des phrases, nous avons augmenté la quantité de données d'entraînement en utilisant un ensemble de données contenant environ 33 milliards de mots. Nous avons utilisé la méthode hiérarchique softmax, une dimension de 1000, et la phrase entière pour le contexte. Nous avons ainsi obtenu un modèle qui a atteint une précision de 72 %. Nous avons obtenu une précision inférieure de 66 % lorsque nous avons réduit la taille de l'ensemble de données de formation à 6 milliards de mots, ce qui suggère que la grande quantité de données de formation est cruciale.

Pour mieux comprendre à quel point les représentations apprises par les différents modèles sont différentes, nous avons inspecté manuellement les plus proches voisins des phrases peu fréquentes en utilisant différents modèles. Le tableau 4 présente un échantillon de cette comparaison. En accord avec les résultats précédents, il semble que les meilleures représentations des phrases soient apprises par un modèle avec softmax hiérarchique et sous-échantillonnage.

5 Additive Compositionality

Nous avons démontré que les représentations des mots et des phrases apprises par le modèle Skip-gram présentent une structure linéaire qui permet d'effectuer un raisonnement analogique précis à l'aide d'une simple arithmétique vectorielle. Il est intéressant de noter que les représentations du modèle Skip-gram présentent un autre type de structure linéaire qui permet de combiner des mots de manière significative par une addition élémentaire de leurs représentations vectorielles. Ce phénomène est illustré dans le tableau 5.

La propriété additive des vecteurs peut être expliquée en inspectant l'objectif de formation. Les vecteurs de mots sont en relation linéaire avec les entrées de la non-linéarité softmax. Comme les vecteurs de mots sont formés pour prédire les mots environnants dans la phrase, les vecteurs peuvent être considérés comme représentant la distribution du contexte dans lequel un mot apparaît. Ces valeurs sont liées logarithmiquement aux probabilités calculées par la couche de sortie, de sorte que la somme de deux vecteurs de mots est liée au produit des deux distributions de contexte.

Le produit fonctionne ici comme la fonction ET : les mots auxquels les deux vecteurs de mots attribuent des probabilités élevées auront une probabilité élevée, et les autres mots auront une probabilité faible. Ainsi, si "Volga River" apparaît fréquemment dans la même phrase avec les mots "Russian" et "river", la somme de ces deux vecteurs de mots donnera un vecteur caractéristique proche du vecteur de "Volga River".

6 Comparison to Published Word Representations

De nombreux auteurs ayant déjà travaillé sur la représentation des mots par des réseaux de neurones ont publié leurs modèles afin de pouvoir les utiliser et les comparer : parmi les auteurs les plus connus figurent Collobert et Weston [2], Turian et al [17], et Mnih et Hinton [10]. Nous avons téléchargé leurs vecteurs de mots sur le web3. Mikolov et al. [8] ont déjà évalué ces représentations de mots sur la tâche d'analogie de mots, où les modèles Skip-gram ont obtenu la meilleure performance avec une marge énorme.

====

Tableau 6 : Exemples de tokens les plus proches donnés par différents modèles connus et par le modèle Skip-gram entraîné sur des phrases utilisant plus de 30 milliards de mots d'entraînement. Une cellule vide signifie que le mot ne fait pas partie du vocabulaire.

Pour donner un meilleur aperçu de la différence de qualité des vecteurs appris, nous fournissons une comparaison empirique en montrant les plus proches voisins des mots peu fréquents dans le tableau 6. Ces exemples montrent que le modèle de grand saut de gramme formé sur un grand corpus surpasse visiblement tous les autres modèles en termes de qualité des représentations apprises. Cela peut être attribué en partie au fait que ce modèle a été entraîné sur environ 30 milliards de mots, ce qui représente environ deux à trois ordres de grandeur de données en plus que la taille typique utilisée dans les travaux antérieurs. Il est intéressant de noter que, bien que l'ensemble d'apprentissage soit beaucoup plus important, le temps d'apprentissage du modèle Skip-gram ne représente qu'une fraction de la complexité temporelle requise par les architectures de modèles précédentes.

7 Conclusion

Ce travail comporte plusieurs contributions essentielles. Nous montrons comment former des représentations distribuées de mots et de phrases avec le modèle Skip-gram et démontrons que ces représentations présentent une structure linéaire qui rend possible un raisonnement analogique précis. Les techniques présentées dans cet article peuvent également être utilisées pour former le modèle de sac de mots continu présenté dans [8].

Nous avons réussi à entraîner des modèles sur plusieurs ordres de grandeur de données par rapport aux modèles publiés précédemment, grâce à l'architecture de modèle efficace en termes de calcul. Il en résulte une grande amélioration de la qualité des représentations des mots et des phrases apprises, en particulier pour les entités rares. Nous avons également constaté que le sous-échantillonnage des mots fréquents permet à la fois une formation plus rapide et une représentation nettement meilleure des mots peu courants. Une autre contribution de notre article est l'algorithme d'échantillonnage négatif, qui est une méthode de formation extrêmement simple qui apprend des représentations précises, en particulier pour les mots fréquents.

Le choix de l'algorithme d'apprentissage et de la sélection des hyperparamètres est une décision spécifique à la tâche, car nous avons constaté que les configurations optimales des hyperparamètres varient selon les problèmes. Dans nos expériences, les décisions les plus cruciales qui affectent les performances sont le choix de l'architecture du modèle, la taille des vecteurs, le taux de sous-échantillonnage et la taille de la fenêtre d'apprentissage.

Un résultat très intéressant de ce travail est que les vecteurs de mots peuvent être combinés de manière significative en utilisant une simple addition vectorielle. Une autre approche de l'apprentissage des représentations de phrases présentée dans cet article consiste à représenter simplement les phrases avec un seul jeton. La combinaison de ces deux approches donne une manière puissante mais simple de représenter des morceaux de texte plus longs, tout en ayant une complexité de calcul minimale. Notre travail peut donc être considéré comme complémentaire à l'approche existante qui tente de représenter les phrases en utilisant des opérations récursives de matrice-vecteur [16].

Nous avons mis à disposition le code d'entraînement des vecteurs de mots et de phrases basé sur les techniques décrites dans cet article sous la forme d'un projet open-source4.