Projet N°2 Science des Données
Word Embeddings et un cas d'usage: la traduction automatique

Eva BOUBA, Meriem DEHMAS, Matthieu SERFATY

Novembre 2020

1 Introduction

Ce rapport présente les résultats de notre travail sur la traduction de mots à partir de plongements lexicaux. La première partie présente les plongements lexicaux de mots et l'usage que l'on peut en faire. Les deuxième et troisième parties présentent les méthodes d'apprentissages supervisé et non supervisé utlisées. La dernière partie présente les résultats obtenus.

2 Les plongements lexicaux et leur lien avec la traduction

Le plongement lexical ("word embedding" en anglais) est une représentation vectorielle d'un mot (ou même d'un ensemble de mots) renfermant une certaine information sur sa sémantique et syntaxe et étant allant une représentation de ce mot. Ces plongements sont pris comme étant les poids de couches de réseaux de neurones entraînés sur des tâches de NLP que l'on pourrait dénommer des "modèles de langues". Les modèles **skip-gram** et **continuous bag-of-words (CBOW)** ainsi que **BERT** (ou encore la déclinaison française camemBERT) en sont des examples.

Les modèles Skip-Gram et CBOW reposent sur le principe que des mots similaires apparaissent dans des contextes similaires. Le premier prédit un contexte en fonction d'un mot tandis que le deuxième effectue l'opération inverse. Dans les deux cas, une couche cachée avec d neurones permet d'obtenir la représentation de chaque mot du dictionnaire par un vecteur de taille d. En entraînant un de ces réseaux sur des corpora mono-langue de tailles suffisamment grandes, on peut donc obtenir la représentation des principaux mots d'une langue.

Ces représentations ont de nombreuses applications dont la traduction. En effet, la traduction revient à entraîner un modèle à associer à chaque vecteur d'un mot source le vecteur de sa traduction. Cette opération pourrait nécessiter des transformations compliquées, mais il a été remarqué que les espaces vectoriels des mots de différentes langues présentent des structures similaires et qu'une simple transformation linéaire est nécessaire.

Le problème de traduction revient alors à trouver la matrice de transformation W permettant d'aligner au mieux les espaces des deux langues.

$$\min_{W \in \mathbb{R}^{d_2 \times d_1}} \sum_i \|W \cdot x_i - y_i\|^2 \tag{1}$$

où (x_i, y_i) est une paire de vecteurs dont les mots source et cible associés (s_i, c_i) ont la même signification et d_1, d_2 sont les dimensions des vecteurs des langues source et cible.

Pour cela, nous utilisons les plongements de FastText, mais par souci de mémoire nous ne prenons que environ les 38000 premiers vecteurs de chaque langue. Cela se doit d'être précisé car lors de l'évaluation de nos traducteurs, seuls les mots présents dans cette portion du vocabulaire peuvent être pris en compte et il est possible que cela améliore nos statistiques en comparaison de celle de l'article de référence([1]) puisque ces mots sont mieux représentés que les mots au delà des 38000 considérés.

3 Traduction supervisée

La traduction supervisée peut s'appliquer dans deux cas de figure :

- Il existe $\mathcal{D} \subset \mathbb{N}$ tel que $\{(s_i, c_i)\}_{i \in \mathcal{D}}$ est connu.
- Il existe $\mathcal{D} \subset \mathbb{N}$ tel que $\forall i \in \mathcal{D}$, s_i et c_i sont homographes.

L'apprentissage se fait alors grâce au dictionnaire $\{(x_i, y_i)\}_{i \in \mathcal{D}}$ en résolvant

$$W^* = \underset{W \in \mathbb{R}^{d_2 \times d_1}}{\operatorname{argmin}} \sum_{i \in \mathcal{D}} \|W \cdot x_i - y_i\|^2 \tag{2}$$

Une fois le vecteur d'un mot x_i projeté, la traduction du mot s'obtient en cherchant le ou les vecteurs de l'espace cible Y les plus proches de Wx_i, ceci revient donc à opérer une instance de *k-plus proches voisins* sur Wx_i dans Y. La traduction de x_i avec précision k (p@k) est considérée comme correcte si la traduction effective du mot se trouve parmi les k mots les plus proches de Wx_i. Dans le cas où un mot se traduirait par plusieurs mots, il suffit que l'un d'eux vérifie cette condition pour que l'on considère la traduction comme correcte. Cette mesure nous donne une première manière d'évaluer un traducteur : l'accuracy de ses traductions en fonction de la précision. Une autre manière de comparer deux traducteurs est de comparer la valeur de la fonction objective donnée dans l'équation 1.

3.1 Apprentissage par descente de gradient

Le problème énoncé peut être résolu par descente de gradient. Nous utilisons pour cela un réseau de neurones à une couche sans fonction d'activation et sans biais représenté sur la figure 1. Les poids du réseau correspondent alors aux coefficients de W. La fonction de perte utilisée est l'erreur quadratique moyenne et la propagation du gradient est faite après un certain nombre d'exemples ce qui équivaut donc à la méthode de descente de gradient stochastique en mini-lots. L'apprentissage est fait sur 60 epochs avec un pas de 0.1 et des lots de 4 exemples.

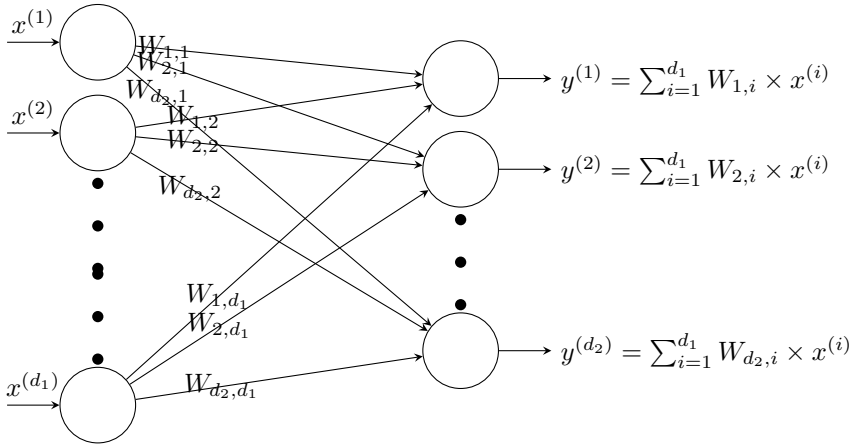

Figure 1: Réseau utilisé pour la descente de gradient

3.2 Apprentissage via un problème de Procuste orthogonal

Il a été montré que l'ajout d'une contrainte d'orthogonalité à la matrice W améliore la traduction obtenue ([3],[4]). Sous cette contrainte, l'équation (1) revient à la résolution d'un problème de Procuste orthogonal pour lequel une solution est obtenue via une décomposition en valeurs singulières (SVD) de la matrice X^TY, i.e:

$$W^* = \underset{W \in \mathbb{R}^{d \times d}, WW^T = I_d}{\operatorname{argmin}} \|WX - Y\|^2 = UV^T; USV^T = SVD(X^TY) \tag{3}$$

3.3 Comparaison des deux méthodes

Nous avons comparé la différence de performance des deux méthodes pour la traduction du français vers l'anglais en fonction de la taille du dictionnaire d'apprentissage. Le dictionnaire $\mathcal{D}$ a été construit avec les mots les plus fréquents dans les corpus d'apprentissage des plongements car ceux-ci sont à priori ceux les mieux représentés par leur plongement.

La seconde méthode étudiée a plusieurs avantages sur la première :

- Elle ne nécessite aucun hyperparamètre.
- L'apprentissage est plus rapide.
- L'orthogonalité de W implique que W^T permet la traduction inverse.
- Elle donne de meilleurs résultats quelque soit la taille de $\mathcal{D}$ (cf. figure 2)

En revanche, son principal inconvénient est que les langues source et cible doivent nécessairement être représentées dans la même dimension.

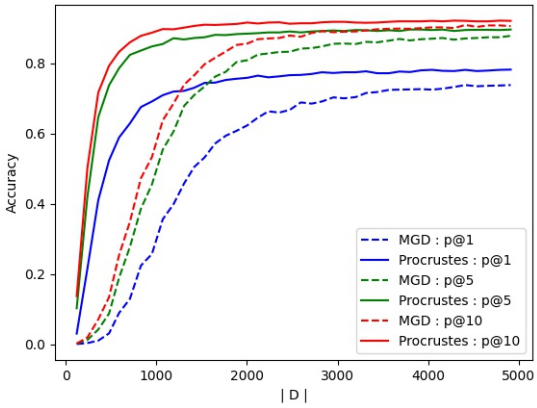

Figure 2: Comparaison des méthodes MGD et Procuste selon la taille de $\mathcal{D}$

4 Traduction non supervisée

La traduction supervisée donne de bons résultats mais son principal inconvénient provient du fait qu'elle soit supervisée. En effet, la traduction d'une langue vers une autre n'est possible que si l'on possède des données parallèles ou si les deux langues ont suffisamment de mots identiques. Or les données parallèles ne sont pas forcément accessibles, et deux langues quelconque n'ont généralement pas beaucoup de mots identiques surtout si elles ne partagent pas le même alphabet. Un autre problème qui se pose dans le deuxième cas est la possible présence de faux-amis. L'utilisation de méthodes non supervisées, c'est à dire ne nécessitant pas de données parallèles, a donc un réel intérêt.

La méthode que nous avons étudiée est le réseau antagoniste génératif. Son principe est de mettre en compétition deux réseaux : le générateur et le discriminateur. Le générateur apprend à générer des données de plus en plus réalistes tandis que le discriminateur apprend à différencier les données réelles de celles générées. Dans notre cas, les données sont des plongements de mots. Le générateur correspond au traducteur, il prend en entrée un vecteur de la langue source et génère une traduction de ce vecteur. Le discrminateur prend en entrée des vecteurs $y \in \mathcal{Y}$ et Wx et donne en sortie la probabilité à ce que le vecteur observé soit une projection (un vecteur Wx) et non un vecteur $y \in \mathcal{Y}$.

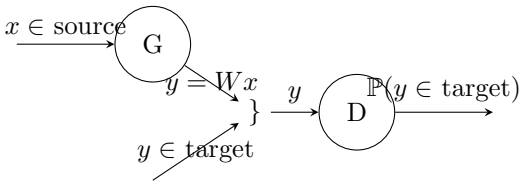

Figure 3: Schéma du GAN utilisé

Comme expliqué dans [1], l'accuracy et la loss ne permettent pas d'évaluer la convergence du réseau.Nous utilisons donc la métrique proposée :

$$\frac{1}{m}\sum_{i=1}^{m} cos(x_i, \underset{y_j, j\leq m}{\text{argmax}}\, CSLS\left(Wx_i, y_j, k\right))$$

où $CLSL(Wx, y, k)$ est la similarité définie dans [1] pour k voisins. Nous avons choisi $m = 1000$ et $k = 10$.

L'architecture du discriminateur est la suivante :

- Dropout de 0.1 et bruit gaussien sur les entrées
- 2 couches cachées de 2048 neurones
- Fonction d'activation LeakyReLu
- Lissage d'étiquettes de 0.2

L'architecture du générateur reste celle présentée sur la figure 1.
Les paramètres d'entraînement des deux modèles sont :

- Optimisation par SGD
- Pas d'apprentissage initial de 0.1
- Learning decay de 0.0008
- Division du pas d'apprentissage après deux diminutions consécutives de la métrique d'évaluation (par 1.05 pour fr→ de, 1.1 pour fr-en et 1.15 pour fr → es)

L'entraînement est fait sur 100 époques avec des lots de 50 exemples. La figure 4 présente la corrélation entre la métrique d'évaluation et l'accuracy des traductions du français vers l'anglais.

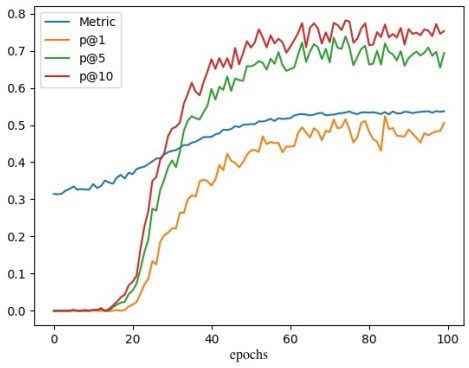

Figure 4: Évolution de la métrique d'évaluation et de l'accuracy des traductions

5 Récapitulatif des résultats

Nous avons testé la traduction entre l'anglais et le français, ains que du français vers l'espagnol et l'allemand. Les résultats obtenus sont dans le tableau 1. MGD et Procuste ont été entraînés avec les fichiers d'entraînement de FastText et les 3 modèles ont été testés sur les fichiers de test.

	fr→en			en→fr			fr→es			fr→de		
	p@1	p@5	p@10	p@1	p@5	p@10	p@1	p@5	p@10	p@1	p@5	p@10
MGD	0.737	0.878	0.909	0.745	0.887	0.911	0.796	0.916	0.940	0.700	0.869	0.906
Procuste	0.780	0.896	0.921	0.788	0.905	0.931	0.819	0.937	0.955	0.753	0.895	0.925
GAN	0.652	0.813	0.847	0.686	0.846	0.849	0.677	0.835	0.868			

Table 1: Résultats des différents traducteurs

6 Références

[1] A.Conneau, G.Lample, M.Ranzato, L.Denoyer, H.Jégou, Word Translation without parallel data.

[2] T.Mikolov et al., Distributed Representations of Words and Phrases and their Compositionality.

[3] C.Xing, D.Wang, C.Liu, Y.Lin, Normalized Word Embedding and Orthogonal Transform for Bilingual Word Translation.

[4] S.Smith, D.Turban, S.Hamblin, N.Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax.