IFT6285 - TP3

Francis de Ladurantaye

December 1st, 2019

1 Modèle Word2Vec

Le modèle Word2Vec consiste en un modèle neuronal peu profond de type perceptron, à une seule couche cachée. L'implémentation du modèle Word2Vec utilisée est celle du *Continuous Bag of Words* (CBOW) fournie par la librairie Gensim. Sous cette variante, l'entraînement consiste à apprendre à prédire un mot d'intérêt à partir des mots qui l'entourent et qui se trouvent dans une certaine fenêtre de contexte (ici les 5 mots avant et après). Les mots du contexte sont envoyés séparément au modèle qui fait la moyenne des vecteurs obtenus suite à leur passage par la couche cachée. Le vecteur des moyennes est ensuite envoyé à la couche de sortie dont le résultat passe par une activation *softmax* afin de produire un vecteur de probabilités que le mot d'intérêt soit chacun des mots du vocabulaire.

Les paramètres contenus dans ce modèle se résument donc à deux matrices, celle de la couche cachée et celle de la couche de sortie. Les mots du vocabulaire étant représentés par des vecteurs *onehot*, le nombre de dimensions de l'entrée et de la sortie de ce modèle doivent donc être de la taille du vocabulaire. Ainsi, le nombre total de paramètres du modèle est :

$$(|vocabulaire| + 1) * hidden\_size + (hidden\_size + 1) * |vocabulaire|$$

où les +1 sont ajoutés pour la prise en compte du biais qu'inclue chaque neurone des couches cachée et de sortie.

2 Modèles entraînés

2.1 Construction du vocabulaire

En vue de calculer le nombre de paramètres du modèle, il est donc nécessaire de connaître la taille du vocabulaire. Au total, deux vocabulaires composés d'unigrammes et de bigrammes ont été construits, avec des seuils respectifs de 50 et 100 occurrences afin d'inclure un unigramme/bigramme. Les entrées du corpus ont été pré-traitées par la méthode `simple_preprocess` offerte par Gensim avec d'être utilisées pour construire les vocabulaires. Le tableau 1 présente le nombre de mots inclut dans chacun des vocabulaires ainsi que les temps requis à leur construction.

Seuil	Nombre de mots	Temps de construction
50	159345	339 secondes
100	149678	343 secondes

Table 1: Nombre de mots des vocabulaires (unigrammes + bigrammmes)

On remarque ici que le temps requis pour construire les vocabulaires n'est pas influencé par le seuil choisi et que ce dernier n'influence, pour les valeurs choisies, que minimalement le nombre

d'unigrammes/bigrammes retenus. Le choix de ces valeurs a été effectué par tâtonnement et a été ajusté suite à plusieurs essais. Pour ne donner qu'un exemple, avec un seuil de 100, le bigramme `green_tea` n'était pas retenu, alors qu'il l'était lorsque le seuil était fixé à 50.

2.2 Entraînement des Word2Vec

Les valeurs sélectionnées pour la taille de la couche cachée, ou taille des vecteurs du modèle, sont 100, 200 et 300. Les temps d'entraînement en secondes se trouvent au tableau 2.

Seuil\Taille couche cachée	100	200	300
50	270	356	422
100	267	353	419

Table 2: Temps d'entraînement des modèles Word2Vec en secondes

On constate de très faibles différences entre les deux seuils comme pour chacun la taille du vocabulaire est sensiblement la même (i.e. $< 1\%$ d'écart). Par contre, le temps d'entraînement augmente largement lorsqu'on augmente le nombre de neurones de la couche cachée, ce qui ne devrait pas nous surprendre compte tenu du fait que doubler la taille de la couche cachée revient approximativement à doubler le nombre de paramètres du modèle. Vous trouverez aux tableaux 3 et **??** le nombre de paramètres de chacun des modèle et l'espace requis pour les stocker sur le disque. Notez que Gensim utilise une précision de 4 octets seulement pour ses modèles afin de réduire l'utilisation mémoire, et que seulement la matrice de la couche cachée est nécessaire au moment de l'inférence.

Seuil\Taille couche cachée	100	200	300
50	32028445	63897545	95766645
100	30085378	60021078	89956778

Table 3: Nombre de paramètres des modèles Word2Vec

Seuil\Taille couche cachée	100	200	300
50	∼122 Mo	∼244 Mo	∼365 Mo
100	∼115 Mo	∼229 Mo	∼343 Mo

Table 4: Taille sur le disque des modèles Word2Vec

3 Sélection des voisins

3.1 Critères de sélection

La méthode utilisée pour la sélection des voisins a été de d'abord réduire la taille du vocabulaire considéré en utilisant la méthode `resize` offerte par la classe `Vector` de Spacy. Celle-ci permet de redimensionner à notre convenance la matrice de poids de la couche caché apprise par nos modèles, permettant du même coup de réduire la taille effective du vocabulaire. L'heuristique utilisée a consisté à utiliser la méthode `resize` afin de réduire le nombre de mot du vocabulaire, en conservant toutefois la taille des vecteurs associés à chacun des mots retenus (i.e. (195000, 300) $\rightarrow$ (30000, 300)). Les réductions considérées ont été celles permettant de conserver 10000, 20000 et 30000 unigrammes/bigrammes uniques.

Cette réduction de la matrice de vecteurs permettait de pouvoir stocker entière en mémoire (12 Go) la matrice des distances entre chacune des paires des mots du vocabulaire. La métrique utilisée pour mesurer la similarité entre deux vecteurs a été la similarité cosinus, qui donne des résultats dans

l'intervalle $[-1, 1]$. Ainsi, lorsque la similarité mesurée est près de 1, cela implique que les deux mots analysés sont très similaires et utilisée de façon quasi interchangeable, alors que lorsqu'elle est près de -1, les mots ne sont jamais utilisés dans les mêmes contextes.

Afin de nous assurer de ne considérer que des mots similaires comme voisins, un seuil de 0,6 pour la similarité cosinus a été fixé afin qu'un mot puisse être considéré comme voisin, et un maximum de voisins 14 ont été affiché pour chacun mot retenus suite au `resize`.

3.2 Analyse des voisins

La première chose que l'on constate en analysant les tables produites selon ces critères et que le nombre moyen de voisins correspondant au critère de seuil minimal de 0,6 est beaucoup plus élevé lorsque 30000 mots sont conservés plutôt que 20000 ou 10000. Cela est tout à fait normal car un plus grand vocabulaire laisse place à avoir un plus grand nombres de mots voisins, mais il faut toutefois en tenir compte pour sélectionner le modèle sur lequel effectuer nos annotations.

Ensuite, on constate que les modèles ayant conservé un grand nombre de mots (30000) et utilisant des vecteurs de plus grandes tailles (200 ou 300) ne contiennent que des mots d'origines étrangères en tête de la table. Cela est probablement dû au fait que leur moins grande fréquence au sein du corpus n'a pas permis de bien les différencier et leurs vecteurs doivent donc se ressembler. Aussi, que cela n'arrive point lorsqu'on limite les vecteurs à une taille de 100 semble renforcer cette hypothèse, car il est en effet plus difficile d'apprendre un plus grand nombre de paramètres avec une faible quantité de données.

La plupart des autres modèles ne contiennent que des noms propres comme premières entrées. Bien que cela soit intéressant lorsqu'il s'agit de choisir le nom d'un enfant à naître, ce n'est pas ce qui nous intéresse ici. Pour les modèles avec des vecteurs de taille 100 et 20000 ou 30000 mots conservés, cela arrête d'être le cas autour de l'entrée 500 à 600. Pour les modèles de taille 200 ayant conservé 20000 mots, cela se produit environ à partir de la 200e entrée, et avant la 50e entrée pour les modèles de taille 300 avec 20000 mots.

Comme les modèles de 10000 mots souffrent du même problème, mais qu'ils ont moins bien performé que les modèles à 20000 mots, nous utiliserons le modèle ayant conservé 20000 mots ayant des vecteurs de 300 valeurs entraînées sur le vocabulaire de seuil 50 pour procéder aux annotations. Le fichier résultant faisant 1,5 Mo, son rétrécissement à 1 Mo a été effectué à l'aide des commandes suivantes, où la seconde permet d'éviter d'avoir la dernière ligne incomplète :

```
truncate --size=1MB out/voisins
```

```
head -n -1 out/voisins > out/temp ; mv out/temp out/voisins
```

4 Annotations

4.1 Sources des étiquettes

La majeure partie des associations entre paires de mots utilisées pour l'étiquetage proviennent du corpus `wordnet` de la librairie NLTK.

Le corpus `Dataset on alternatives` provenant du site http://disi.unitn.it/~bernardi/index.php?page=resources a aussi été utilisé.

Enfin, les données tirées de `wordsim353_agreed.txt` trouvé sur le site http://alfonseca.org/eng/research/wordsim353.html font aussi partie des paires étiquetées auxquelles je me suis référées.

4.1.1 Étiquettes sélectionnées

Les étiquettes utilisées pour les annotations sont les suivantes :

- SYNONYM : Mot de sens similaire
- ANTONYM : Mot de sens opposé
- PARTOF : Méronyme (est partie de ce qui est représenté par l'autre mot)
- MADEOF : Holonyme (contient ce qui est représenté par l'autre mot)
- COHYPO : Hyperonyme (moins spécifique, inclut le sens de l'autre mot)
- COHYPO : Cohyponyme (a au moins un hyperonyme en commun, sans être un synonyme ou un antonyme)
- HYPO : Hyponyme (plus spécifique, est inclut dans le sens de l'autre mot)
- RELATED : Mot lié non représenté par les étiquettes ci-dessus
- MORPHO : Morphologie (variante orthographique du mot)

Sur un total de 1366 mots ayant au moins deux voisins annotés, voici la répartition des étiquettes:

Annotation	Nombre	Pourcentage
SYNONYM	198	4,50%
ANTONYM	44	1,00%
PARTOF	75	1,71%
MADEOF	40	0,91%
HYPER	189	4,30%
COHYPO	1731	39,36%
HYPO	282	6,41%
RELATED	1	0,02%
MORPHO	1838	41,79%
TOTAL	4398	100,00%

Table 5: Fréquence par étiquette sur les mots ayant au moins deux voisins annotés

De plus, 3 voisins avaient 3 annotations, 103 voisins avaient 2 annotations et 4292 voisins n'avaient qu'une annotation.