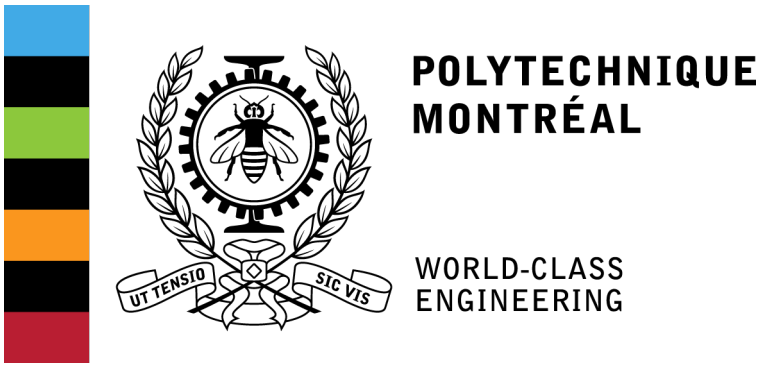
École Polytechnique de Montréal

Département de génie informatique et génie logiciel



INF6953I - Fouille de données

RAPPORT DU LABORATOIRE 1

Foromo Daniel Soromou 1759116

Roger Lobe 1679117

Gilles Eric Zagre 1146014

Chargé de laboratoire

Alexandre dos Santos

21 Mai 2018

Table des matières

1 Introduction **2**

2 Question 1 : Description des méthodes d'extraction d'attributs **2**
2.1 Approche Bag-of-Word 2
2.2 Approche Doc2Vec 2

3 Question 2 : Comparaison des méthodes **3**

4 Question 3 : Bi-gram **3**

5 Question 4 : Resultats de la classification avec BoW **3**

6 Question 5 : Resultats de la classification avec Doc2Vec **9**

7 Références **14**

1 Introduction

Dans ce laboratoire nous allons tester différentes méthodes d'extraction d'attributs d'un document texte (collection de mots) dans le but d'en faire une classification. Pour ce faire nous allons nous baser sur un exeemple de classification pour le dataset ImDB Sentiment de Stanford. Nous nous interresserons plus précisement à la méthode *Bag-of-Words* (BoW) et à' l'une des méthodes de représentations vectorielles *Doc2Vec*. Pour chacune de ces méthodes nous explorerons plusieurs configurations (avec ou sans *stemming*, *tokenizer*...)

2 Question 1 : Description des méthodes d'extraction d'attributs

2.1 Approche Bag-of-Word

Sachant que la plupart des algorithmes d'apprentissage (et de classification) prennent en entrée les vecteurs de même dimension, l'objectif de la méthode Bag-of-Word (BoW) est de transformer chaque document en un vecteur dont la taille est celle d'un vocabulaire prédéfini |V| et qui contient des statistiques sur la fréquence d'apparition des mots du vocabulaire par exemple. L'ensemble des documents formant le corpus sera ainsi représenté par une matrice composée des vecteurs précédemment constitués de taille |V| et qui seront donc utilisés comme entrées de nos algorithmes de classification.

En résumdès que les données textes sont collectées, nettoyées et que le corpus est ainsi constitué, la méthode BoW consiste à :

(i) Concevoir le vocabulaire : On liste tous les différents mots présents dans le corpus.

(ii) Créer des vecteurs de documents : Pour chaque document à l'étude, on crée un vecteur de taille N pour référencer les N mots du vocabulaire. À chaque composante du vecteur, soit on affecte une valeur booléenne (1 pour indiquer la présence du mot dans le vocabulaire et 0 sinon) pour une représentation binaire. Soit on inscrit le nombre de fois que le mot apparaît dans le document ou alors on utilise la fréquence d'apparition du mot dans le document par rapport aux autres mots du même document.

Cette approche est éfficace et simple, mais on perd de l'information sur l'ordre des mots et donc de línformation sur la signification. De plus, l'usage d'un vocabulaire prédéfini peut s'averer limitatif s'il est trop petit car plusieurs mots ne seront pas classés. Si le vocabulaire utilisé est trés grand on se retrouve avec des matrices éparses contenant plusieurs zeros qui sont difficiles à traiter.

2.2 Approche Doc2Vec

L'approche Doc2Vec transforme aussi des documents en vecteurs de taille fixe. C'est une méthode qui est inspirée de la méthode Word2Vec pour la représentation vectorielle des mots. Tandis que cette dernière prédit les mots environnants, Doc2Vec vise à prédire le contexte ou les paragraphes dans un document grâce a un réseau de neurones. Cette méthode comporte à la base deux étapes essentielles :

(i) l'étape de la formation : Après la collecte de tous les documents, un vecteur d'attribut de taille fixe est généré pour caractériser chaque document par

apprentissage non supervisé à l'aide d'un réseau de neurones. Le modèle est entrainé pour effecturer l'extraction des attributs.

(ii) l'étape d'inférence : pour un nouveau document peut être présenté, les poids troués lors de l'entra[inement sont utilisés pour calculer le vecteur d'attributs du document.

3 Question 2 : Comparaison des méthodes

Commen mentionné précedemment, BoW est simple mais fait perdre de l'information sur l'ordre des mots et la signification du texte. De plus, peut s'averer limitatif pour decrire un texte si le vocabulaire est trop petit car plusieurs mots ne seront pas classés. Si le vocabulaire utilisé est trés grand on se retrouve avec des matrices éparses contenant plusieurs zeros qui sont difficiles à traiter. La méthode Doc2Vec génère des vecteurs de faible dimension par rapport aux vecteurs issus de l'approche BoW. Contrairement à BoW, Doc2Vec n'ignore pas le contexte des mots. Or le contexte d'un mot dans une phrase caractérise le mot sur l'aspect syntaxique et sur son aspect sémantique. En considérant par exemple la phrase « je suis en train de conduire » le mot «auto» correspond mieux à cette phrase que le mot «fourmis».

— Doc2Vec tient compte de l'ordre des mots contrairement à BoW.
— Doc2Vec permet une généralisation à des documents plus longs et peut apprendre à exploiter les données non étiquetées.

Toutefois, avec Doc2Vec il peut aussi etre difficile de savoir quelle est la taille optimale du vecteur d'attributs et d'ajuster les options particulièrement pour l'inférence. Pour obtenir une bonne inférence il est necessaire de bien entra[iner le modéle et donc de disposer d'une très grande quantité de documents pour un apprentissage non supervisé. De plus l'interprétation des attributs obtenus (features) n'est pas nécessairement intuitive.

4 Question 3 : Bi-gram

Un bigram est un groupe de deux mots consécutifs pris parmi les mots du corpus formé (ou N-Gram pour N mots consécutifs). Pour comparer par exemple des différents champs lexicaux, on a besoin d'un certain indicateur. Il existe dans un texte et par extension dans le langage une forme de dépendance plus ou moins grande entre les mots. Le pronom "je" par exemple a de fortes chances d'être suivi par un verbe. En assignant une certaine probabilité d'apparition à chaque bigram, on peut savoir si le couple de mots peut se mettre ensemble et celà peut fortement améliorer les performances lors d'une recherche de similarité syntaxique par exemple.

5 Question 4 : Resultats de la classification avec BoW

Pour l'approche BoW, nous avons effectué des tests pour chacune des configurations avec des tailles de vocabulaire différents. Ainsi pour cette approche, nous avons 8 configurations au totale. Ci-dessous se trouve la liste des configurations.

1. Config 1 : "use_bigram" :False, "use_nltk_tok" :False, "use_stemmer" :False
2. Config 2 : "use_bigram" :False, "use_nltk_tok" :False, "use_stemmer" :True
3. Config 3 : "use_bigram" :False, "use_nltk_tok" :True, "use_stemmer" :False
4. Config 4 : "use_bigram" :False, "use_nltk_tok" :True, "use_stemmer" :True
5. Config 5 : "use_bigram" :True, "use_nltk_tok" :False, "use_stemmer" :False
6. Config 6 : "use_bigram" :True, "use_nltk_tok" :False, "use_stemmer" :True
7. Config 7 : "use_bigram" :True, "use_nltk_tok" :True, "use_stemmer" :False
8. Config 8 : "use_bigram" :True, "use_nltk_tok" :True, "use_stemmer" :True

Pour chacune des configurations ci-dessus, les figures suivantes illustrent la performance de classification en fonction de la taille du vocabulaire.

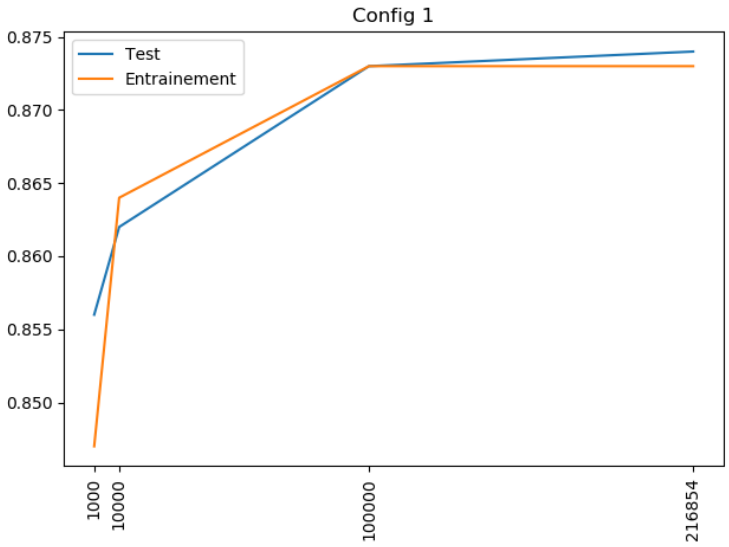


FIGURE 1 – Performance vs taille du vocabulaire pour BoW avec Config 1

En observant les différents graphiques, on remarque que quand on augmente la taille du vocabulaire, la précision du modèle augemente aussi. De plus, on constate que la configuration 6 nous donnes une meilleure performance en test de 89.9% pour le cas où la taille de notre vocabulaire est 216854. Finalement, les différents résultats nous montres que la combinaison de l'option bigram et stemmer à vrai permet d'obtenir de meilleures résultats.

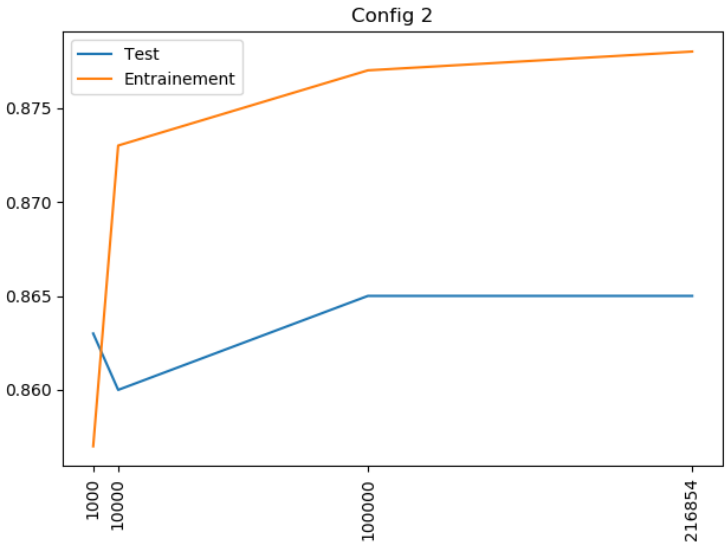


FIGURE 2 – Performance vs taille du vocabulaire pour BoW avec Config 2

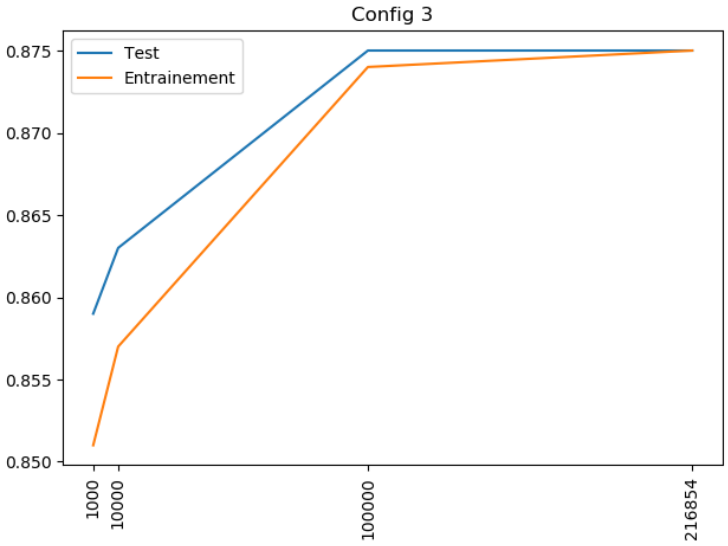


FIGURE 3 – Performance vs taille du vocabulaire pour BoW avec Config 3

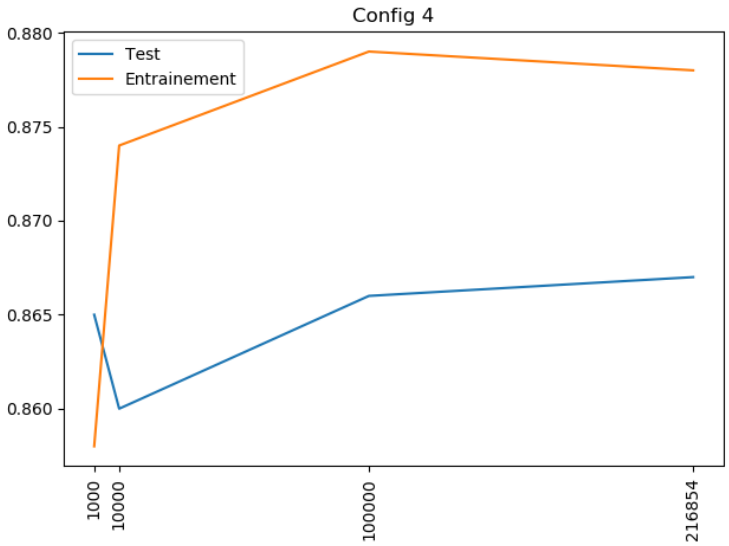


FIGURE 4 – Performance vs taille du vocabulaire pour BoW avec Config 4

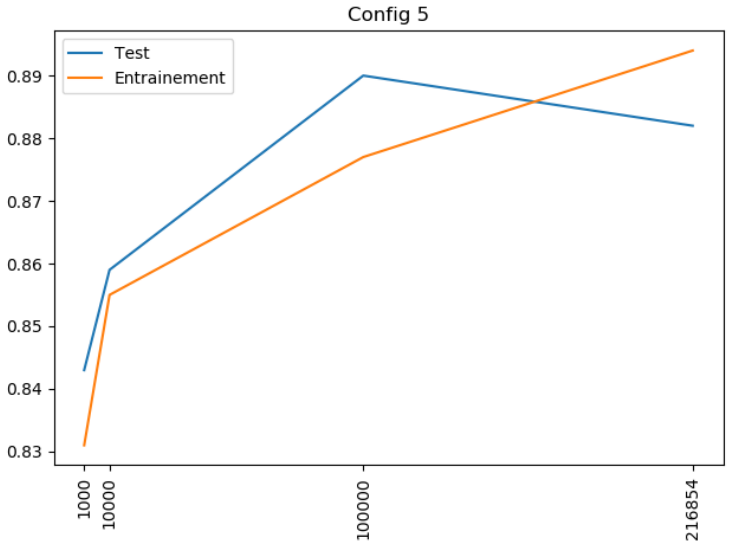


FIGURE 5 – Performance vs taille du vocabulaire pour BoW avec Config 5

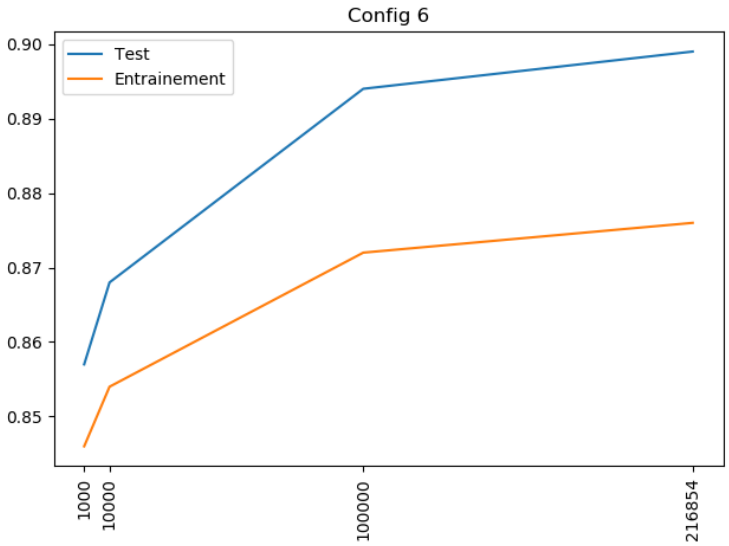


FIGURE 6 – Performance vs taille du vocabulaire pour BoW avec Config 6

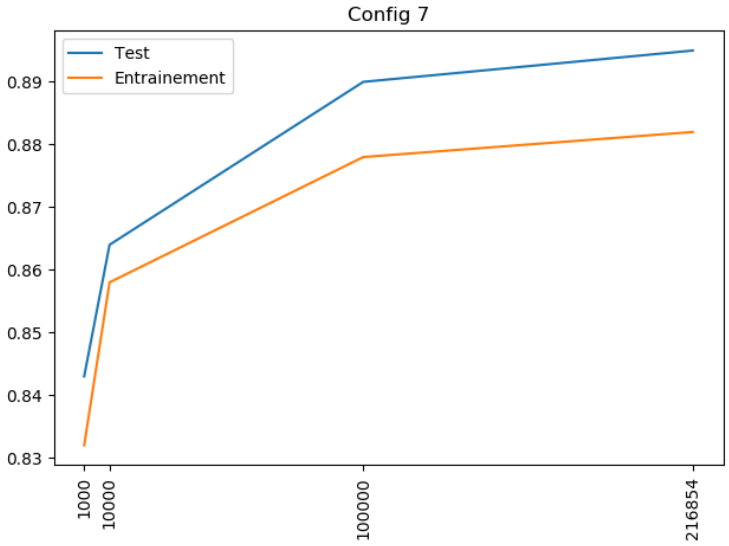


FIGURE 7 – Performance vs taille du vocabulaire pour BoW avec Config 7

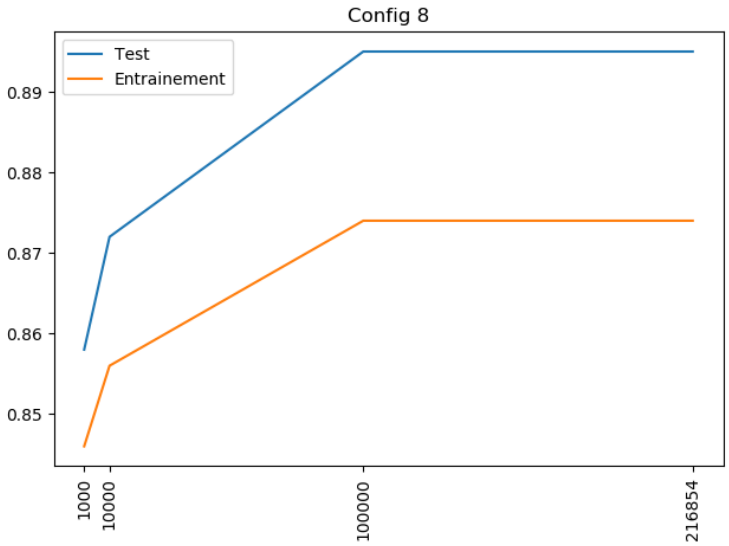


FIGURE 8 – Performance vs taille du vocabulaire pour BoW avec Config 8

6 Question 5 : Resultats de la classification avec Doc2Vec

Pour l'approche Doc2Vec, nous avons effectué des tests pour plusieurs configurations avec des tailles de vecteurs différents pour chaque configuration. Ainsi pour cette approche, nous avons 8 configurations au totale. Ci-dessous se trouve la liste des configurations utilisées :

1. Config 1 : "use_bigram" :False, "use_nltk_tok" :False, "use_stemmer" :False
2. Config 2 : "use_bigram" :False, "use_nltk_tok" :False, "use_stemmer" :True
3. Config 3 : "use_bigram" :False, "use_nltk_tok" :True, "use_stemmer" :False
4. Config 4 : "use_bigram" :False, "use_nltk_tok" :True, "use_stemmer" :True
5. Config 5 : "use_bigram" :True, "use_nltk_tok" :False, "use_stemmer" :False
6. Config 6 : "use_bigram" :True, "use_nltk_tok" :False, "use_stemmer" :True
7. Config 7 : "use_bigram" :True, "use_nltk_tok" :True, "use_stemmer" :False
8. Config 8 : "use_bigram" :True, "use_nltk_tok" :True, "use_stemmer" :True

Pour chacune des configurations ci-dessus, les figures suivantes illustrent la performance de classification en fonction de la taille du vecteur d'attributs.

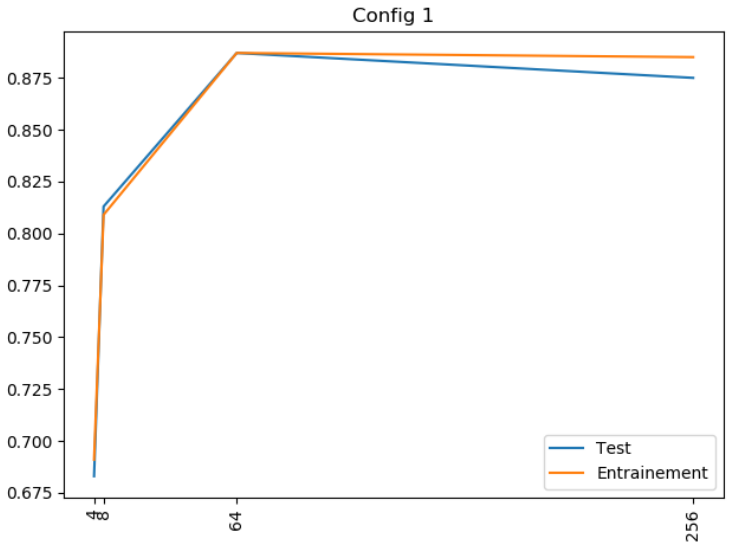


FIGURE 9 – Performance vs taille du vecteur pour Doc2Vec avec Config 1

On remarque que la taille optimale du vecteur d'attributs est autour de 64. La performance moyenne en test est autour de 88% et la meilleure configuration est obtenue avec la configuration 5 (89,68%) mais la différence n'est pas significative par rapport à d'autres configurations. Les autres options (bigram, stemmer, tokenize) ne semblent pas avoir un impact majeur quand on utilise la méthode Doc2Vec.

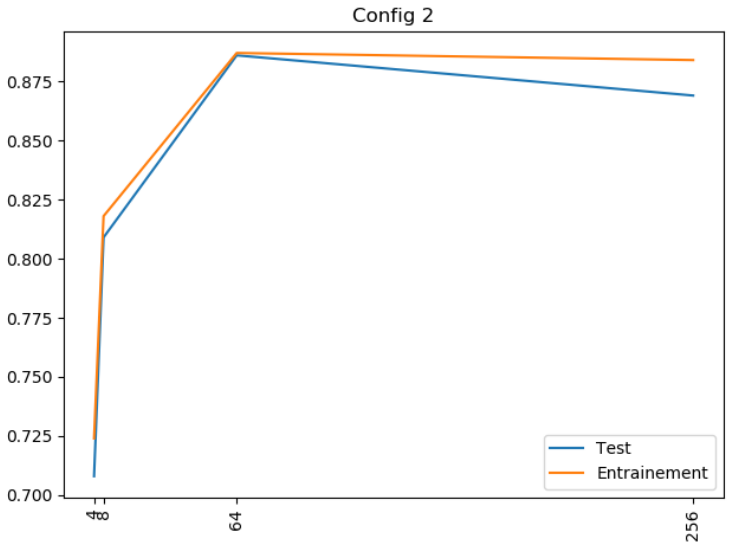


FIGURE 10 – Performance vs taille du vecteur pour Doc2Vec avec Config 2

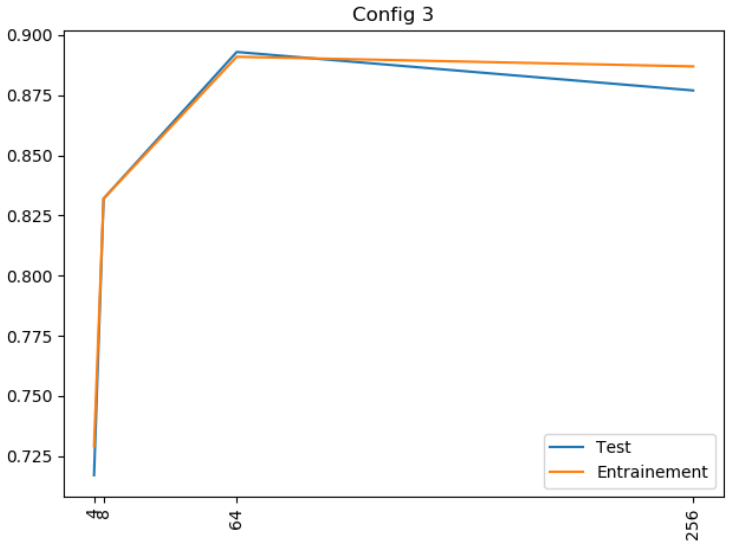


FIGURE 11 – Performance vs taille du vecteur pour Doc2Vec avec Config 3

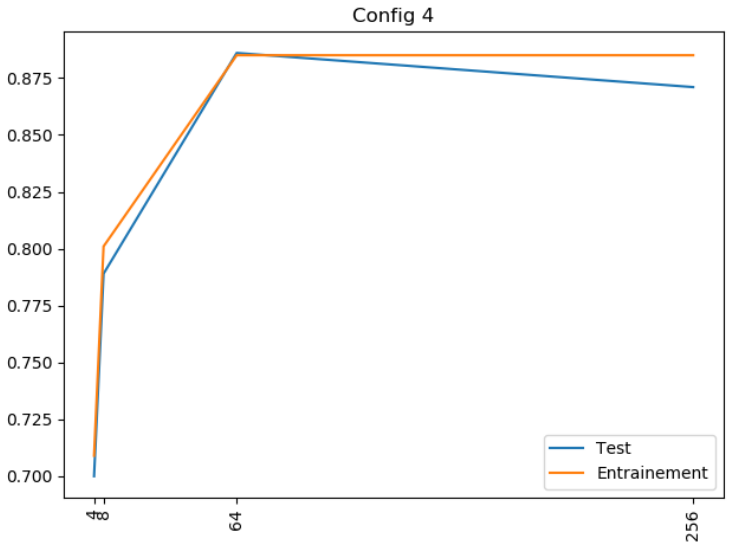


FIGURE 12 – Performance vs taille du vecteur pour Doc2Vec avec Config 4

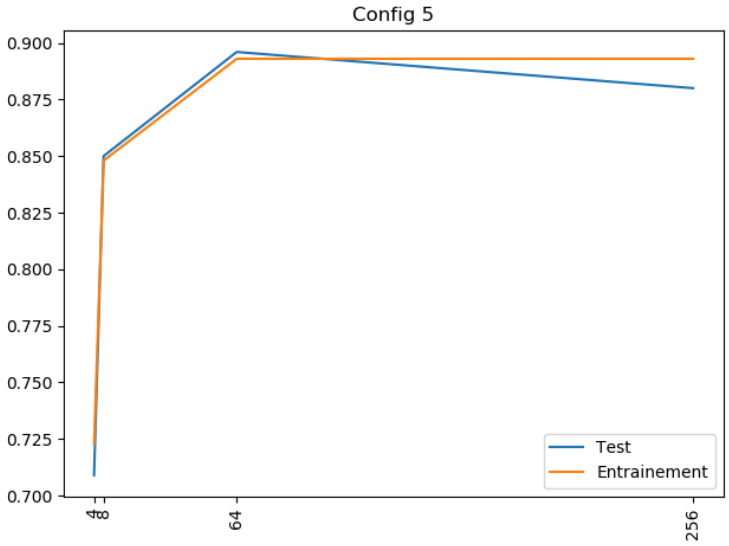


FIGURE 13 – Performance vs taille du vecteur pour Doc2Vec avec Config 5

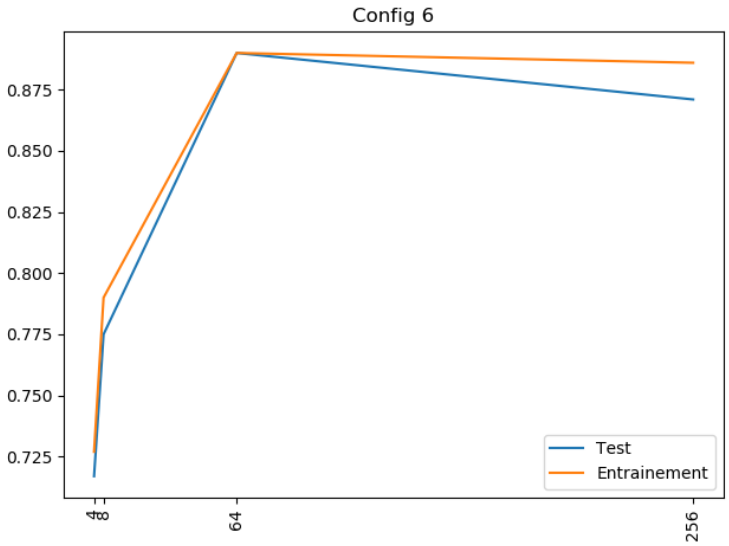


FIGURE 14 – Performance vs taille du vecteur pour Doc2Vec avec Config 6

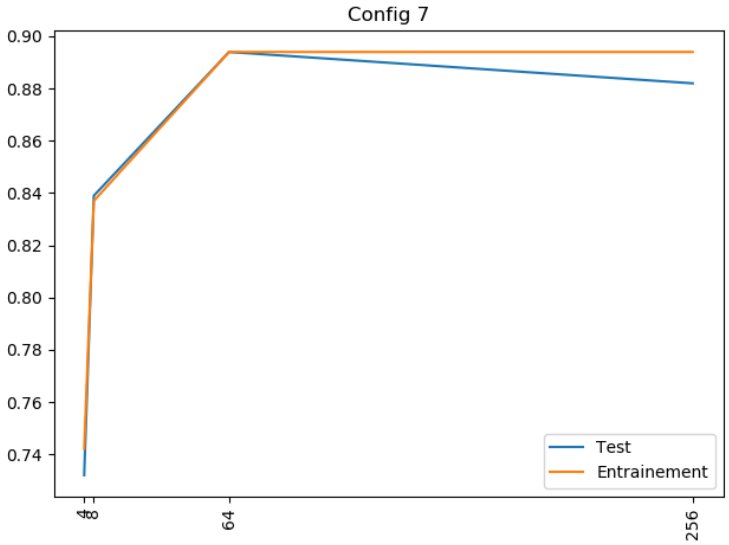


FIGURE 15 – Performance vs taille du vecteur pour Doc2Vec avec Config 7

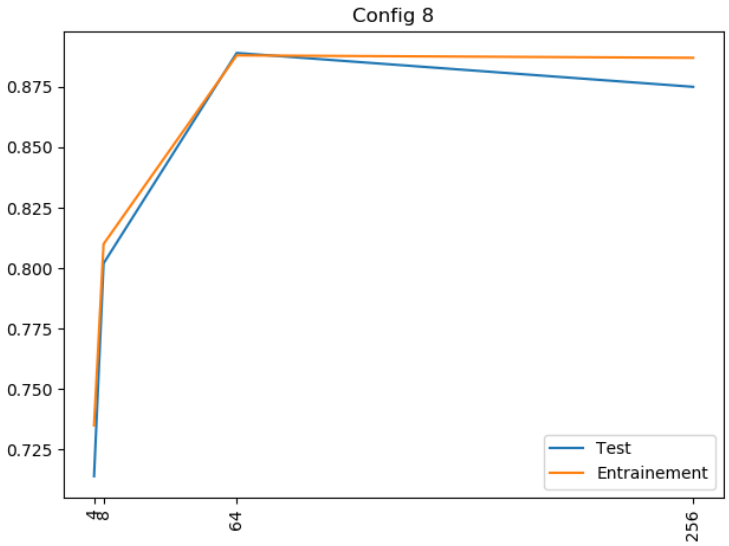


FIGURE 16 – Performance vs taille du vecteur pour Doc2Vec avec Config 8

7 Références

Références

[1] Notes de cours INF6953I École Polytechnique de Montréal.

[2] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space, [cs.CL] 7 Sep 2013.

[3] Quoc Le, Tomas Mikolov. Distributed Representations of Sentences and Documents, [cs.CL] 22 May 2014