Big Data Mining - Assignment

Dimitrios Vogiatzis
CS2.18.0004

Victor Giannakouris
CS3.18.0002

1 Abstract

In this report we describe in detail the techniques that we used in the assignment of the Big Data Mining course. In summary, given an news input dataset the main goals of this assignment were to generate a Wordcloud for each possible category, duplicate detection with respect to the cosine distance metric, the implementation and evaluation of several state-of-the-art classification algorithms, as well as a custom architecture that overperforms the aforementioned state-of-the-art algorithms in terms of a number of evaluation metrics. Through experimental evaluation we showcase that our architecture achieves better performance for all of the evaluation metrics that we have used, including accuracy, precision, recall, ROC and F-Measure.

2 Introduction

2.1 Goals

The goal of this assignment is the development of a system that will meet all the requirements by implementing modules for the following: 1. Generation of a Wordcloud for each of the possible news categories, 2. Detection of articles with high degree of similarity between them (duplicate detection), 3. The implementation and evaluation of several state-of-the-art classification algorithms and 4. The development of a custom architecture that will outperform the algorithms defined in 3.

2.2 Installation

3 Implementation

3.1 Wordcloud

3.2 Duplicates Detection

3.3 Classification Implementation

In this section we describe in detail the libraries that we used in order to implement the required classification algorithms, vectorizers, as well as dimensionality reduction modules.

3.3.1 Vectorizers

Two vectorizers were used for the assignment's purposes, that is, a bag-of-words (BOW) model and the Word2Vec (W2V) model.

Bag-of-Words. The bag-of-words[1] model is one of the simplest approaches used in text mining. A document is represented as a n-sized vector, where n the size of the dictionary. An element at the position i of a vector X represents the frequency of the i^{th} word of the dictionary in the vector X, where i denotes the word index. We used the *CountVectorizer* instance of scikit-learn library for leveraging the Bag-of-Words model. Below there is a sample code snippet.

```
from sklearn.feature_extraction.text import CountVectorizer

input_docs = open("docs.txt").readlines()
cv = CountVectorizer()
cv.fit(input_docs)
```

Word2Vec. Word2Vec[2] is a more complex model. In summary, Word2Vec is a two-layer neural network trained to reconstruct linguistic contexts of words, which are also called *word embeddings*[3]. In this model each word is represented as a vector of numbers, in contrast with conventional models like TF-IDF where each word is represented as a single weight number. The main benefit of representing words as vectors is that different words with the same meaning will be close in the vector space, i.e. the word "king" will be close to the word "queen".

To implement Word2Vec we used the gensim[1] library

3.4 Neural Network Architecture

References

[1] Bag-of-Words. https://en.wikipedia.org/wiki/Bag-of-words_model.

[1] `https://radimrehurek.com/gensim/`

[2] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[3] Word embedding. https://en.wikipedia.org/wiki/Word_embedding.