ChatEGP Project: Newspaper Sentiment Analysis

Philopateer Amgad, Youssef Amin, and George Sherif

May 8, 2023

Abstract

Sentiment analysis is one of the most popular topics in Natural Language Processing (NLP). The aim of doing sentiment analysis is to determine people's emotions towards a particular topic, whether positive, neutral, or negative. In this project, we aim to develop a sentiment analysis model that determines the readers' sentiment toward Arabic financial news.

We divided the implementation into two parts. The first part includes training and fine-tuning the BERT model with an English dataset that consists of over 4500 headlines for financial news. After that, we evaluated the model using well-known metrics such as accuracy and f1-score. The second stage is translating a considerably smaller manually labeled Arabic dataset into English. The translated headlines are used to evaluate the model's performance in determining the sentiment of financial news headlines from another region.

Prior to each part discussed earlier, the texts used are processed to remove stop words, punctuation, and other non-relevant information. These texts are provided to BERT, which works by using a deep neural network to process and understand the meaning of the text. We decided to use BERT as our architecture as it has shown promising results in various NLP tasks, including sentiment analysis.

The final model allows us to understand the public sentiment toward finance news topics, which can be valuable for identifying some economic or financial trends at a given time. Additionally, this project highlights whether the news sentiment from the Arabic region can be identified using a model fine-tuned with news from a different region.

1 Introduction

Natural Language Processing (NLP) is the field where we aim to make computers understand natural languages, not only by reading or processing them but also by understanding the meaning behind them, and therefore be able to perform various tasks such as topic modeling, language modeling, sentiment analysis, and others. Nowadays, NLP is a hot topic of research, where scientists try to improve the complexities of the language models and enhance their training by using bigger and better datasets representing real life. An example of this could be the more powerful version of GPT-4 than its predecessor GPT-3, which was trained using much more parameters and larger datasets, making it able to perform tasks that are currently beyond the capabilities of GPT-3.

Our project focuses on sentiment analysis, one of several NLP applications. Sentiment analysis is the process of categorizing the sentiment of the person authoring the text by analyzing his feelings through words and categorizing the sentence as positive, neutral, or negative. We attempt to determine the sentiment of financial Arabic newspaper articles for this project. This issue is critical because it may serve as an indicator of the economy, financial status, or behavior during a certain time period. The model performing the sentiment analysis could be used to detect any difficulties facing that specific sector that yielded negative sentiments in its news.

To be able to perform the sentiment analysis, the project was divided into several phases. The first step was to analyze the dataset, in order to have an insight into the data we are working on. Data analysis allowed us to figure out the important features of the data to keep and the non-important ones to remove. This was feasible through the availability of a wide variety of libraries, such as *matplotlib*, *seaborn*, and the *tf-idf* [SJ04].

The step following the data analysis was to actually preprocess the data. During this phase, the *nltk* [BK09] library was used, which offered us several functions that perform the cleaning process. The preprocessing phase included tokenizing the texts, removing stop words, stemming, and standardizing

the case of the texts to be all lowercase. The tokens are then appended again into sentence form to be in the correct format for the BERT tokenizer in the following step.

The third phase was to choose the architecture of our project and to train the model. We chose the BERT architecture [DCLT18], since it is a powerful tool that can perform several NLP tasks with high accuracy, including sentiment analysis. At first, we fine-tuned the BERT model with an English dataset, that is very similar to what we wanted to do with our Arabic dataset. The following step was to translate the Arabic text files using MarianMTModel [JDGD+18], which is a Neural Machine Translation framework written in C++ language. These translated text files are then used to evaluate the effectiveness of the model to determine their sentiment, and whether the translation will affect the accuracy or not.

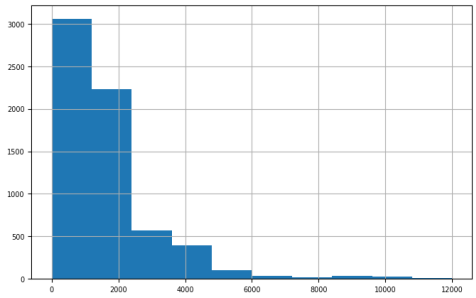
2 Data Analysis

Our chosen dataset is called SANAD (Single-Label Arabic News Articles Dataset). The articles were collected using Python scripts written specifically for three popular news websites: AlKhaleej, AlArabiya and Akhbarona. Articles are categorized into 7 topics as shown in Figure 1.

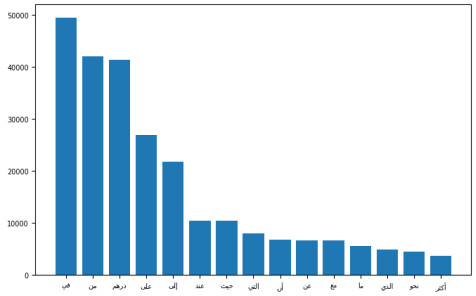
s0.4s

Figure 1: Complete Arabic dataset distribution

Our focus is solely on the Finance category. In the following lines, we aim to demonstrate our knowledge of the dataset through analysis and visualizations. Our corpus consists of 6500 individual text files. The distribution of the length of each file in characters is displayed in Figure 4(a), showing that almost half of the dataset has a length of 1000 characters. The median of word lengths is 5. A very important application of data analysis is determining stop words. The library nltk [BK09] has some universal Arabic stop words. These are mainly propositions (حروف الجر و اسماء الوصل) and punctuation. These of which appearing in our dataset are displayed in 2(b). This is consistent with the data obtained from calculating Term Frequency(tf), as many of the highest scoring words in the tf table 1 are those which are in the nltk list.



((a)) Length of files in Characters

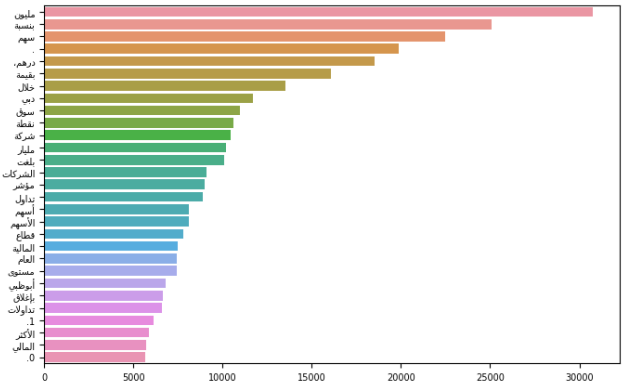


((b)) Frequency of Universal Stop Words in our dataset

Table 1: TFs

Word	في	من	على	إلى
Sum of TFs	90.61	68.17	41.14	33.79

Another representation of the most common words is the word cloud as in figure 4(b). Here we can see a more domain-oriented possibility for stop words removal. Words of currencies (دينار,درهم), stocks سهم or orders of magnitude (مليار,مليون) could be regarded as stop words in the domain of finance. These were obtained after removing the universal stop words and evaluating the tf once more. Using bigrams could also be inferential [WM12]. Using the library sci-kit learn, the method CounVectorizer [PVG+11] is used to calculate bigrams and trigrams. Using this data, we are able



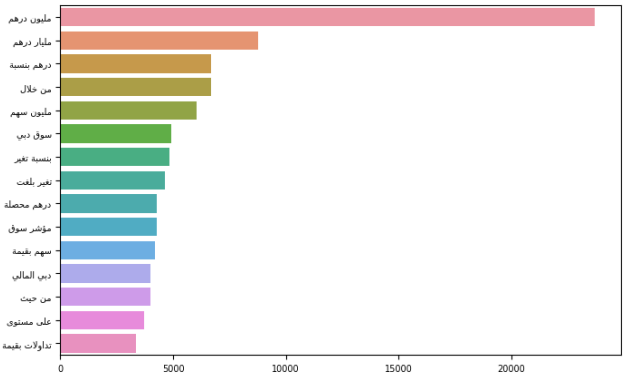
((a)) Domain Specific Stop words

((b)) Word cloud

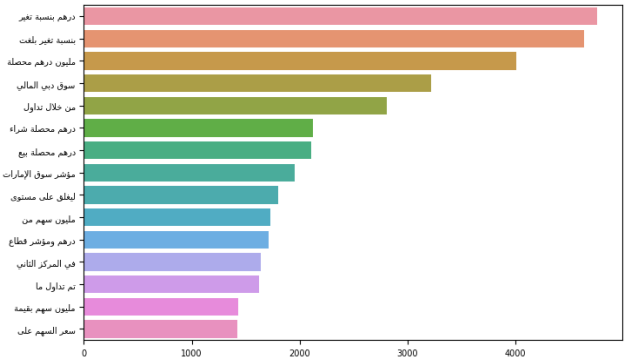
Table 2: TF-IDFs

Word	بلغت	تداول	بقيمة	بنسبة
Sum of TF-IDFs	201.05	213.18	338.13	481.95

to make more informed decisions in preprocessing and later in hyper-parameter tuning. As a simple example, another word cloud is shown 5 highlighting the most significant words after removing some of the stop words. Which is also consistent with the results obtained from tf-idf. The tf-idf enhances data obtained from the tf, as many of the words with a high tf score are insignificant (stop words).



((a)) Bigram frequencies



((b)) Trigram frequencies

Figure 4: N-grams

3 Data Preprocessing

Preprocessing of the data is the part concerned with cleaning the data. By cleaning the data we mean removing any unnecessary information that may result in confusing our model. To do this part we used the help of the NLTK library [BK09], which is a very well-known library that has lots of utilities that we can use. The first step we did is to tokenize the sentence coming from the dataset using the *nltk.word_tokenize()* function, in order to be easily able to work on the word level. The second step was doing the stemming part which is concerned with returning words to their origin, and also applying lowercase to all words in a sentence. For this part, we used the *PorterStemmer()* function. Finally, we removed stop words and punctuation using the set of stop words and punctuation known in the English dictionary, the reason we applied this step is that those stop words are not of great impact on the sentence, so we removed them to consider only words that matter.

Figure 5: Word Cloud after removing some stop words

References

[BK09] Edward Loper Bird, Steven and Ewan Klein. *Natural Language Processing with Python. O'Reilly Media Inc.* 2009.

[DCLT18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[JDGD+18] Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. Marian: Fast neural machine translation in C++, July 2018.

[PVG+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[SJ04] Karen Spärck Jones. Idf term weighting and ir research lessons. *Journal of documentation*, 60(5):521–523, 2004.

[WM12] Sida I Wang and Christopher D Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 90–94, 2012.