Predict the sentiment of a tweet and the words supporting the sentiment
Machine Learning for Natural Language Processing 2021

El Atifi Sofian
ENSAE
sofian.el.atifi@ensae.fr

Mignon Hadrien
ENSAE
hadrien.mignon@ensae.fr

1 Problem Framing

Why does BERT do better than simple "bag of words" methods at predicting the sentiment of a tweet? In order to answer this question, we propose to perform two supervised learning tasks (The link of the notebook is in footnote bellow[1]):

- Task 1 consists in comparing the performance of several embeddings to predict the sentiment of a tweet
- Task 2 consists in predicting the words that support this sentiment.

In the first task, we compare and explain the results of 4 different types of embeddings:

1. the term frequency based vectorizer in each document (CountVectorizer)
2. the document vectorizer based on the Term-Frequency - Inverse Document Frequency (TF-IDF)
3. the TF-IDF vectorizer where a term corresponds to 2 tokens instead of one (ngram = 2)
4. the pre-trained neural network vectorizer called BERT to which we add a learning layer from the original data of our study.

The first three methods are "bag-of-words" methods because the vector of each word (token) does not take into account the order of the words (tokens) nor the context of the words. On the contrary, the BERT model has been pre-trained to take into account the context(Jacob Devlin, 2019).

We will perform two supervised statistical learning tasks. The first task simply aims to predict the sentiment of a tweet (neutral, positive or negative). The second task aims to predict the words that support the sentiment of the tweet. In the first task, we are interested in comparing the prediction performance of the different embeddings. In the second task, we only use BERT and we focus our study on the design of the pipeline that leads from the training data to the prediction through the transformation of the embeddings.

2 Experiments Protocol

The data used is taken from a Kaggle competition[2]. We have about 23000 tweets which are all labeled with either a positive, negative or neutral sentiment and with the supporting words. For computational purposes, we sample 10% and 1.5% of the dataset for task 1 and task 2 respectiviely. In task 1, we apply a specific preprocessing for the "bag of words embeddings" because they do not rely on a pre-existing vocabulary. We therefore try to keep as much relevant information as possible: each hashtag (#), each emoji, each emoticon, each punctuation mark and the first 5 characters of a url address (https) have their own token. In the case of BERT, we do not do any preprocessing. All words and symbols that are not listed in the BERT vocabulary are automatically associated with the token "UNKNOWN".

In task 1, since we want to compare the performance of the embeddings and not the prediction models, we always apply the same SVM model for each bag of word embeddings. What changes is only the input word vectors, not the model.

The modeling from BERT follows the following four steps:

1) we retrieve the pre-trained embeddings for each word.

[1] Notebook Colab link here: https://drive.google.com/file/d/1eUpvTOPTbdVX03BZ1sx6nJlrZw65LctR2/view?usp=sharing

[2] Competition:https://www.kaggle.com/c/tweet-sentiment-extraction/overview and Data:https://www.kaggle.com/c/tweet-sentiment-extraction2/overview

2) we add a layer to the BERT network. This means that we modify the pre-trained embeddings to take into account the specificity of the studied corpus by using the BERT neural network.

3) we create new word embeddings to have a vector for each document

4) we add a linear layer on the document vectors to predict a probability to belong to each class (3 probabilities for 3 classes).

In task 2, we no longer want to predict a probability per sentiment but rather a probability per token. So we modify the 4th step: instead of retrieving a probability vector of size 3, we retrieve a probability vector of size 70, that is the number of tokens in each tweet. The label of the target also changes: the label is no longer the sentiment of the tweet but the supporting words of the tweet. We thus created a boolean vector of size 70 for each tweet. Target = $(x_1...x_i...x_{70})$ with $i = 1$ if the i-th word is a support word, 0 otherwise.

All task 1 models are trained on the same 2473 tweets and tested on the same 275 tweets (10% of the total). We compute the balanced accuracy of each of these methods to compare their respective performances. We then perform a qualitative analysis of these four models in order to understand in detail why some of them correctly predict the sentiment of a tweet while others are wrong.

We encountered computational problems on task 2 and simply trained and tested the model on a small number of tweets (n_train = 370, n test = 21). But for task 2, model performance was not the main goal. Since this second task is not standard, its interest is mainly pedagogical since it allowed us to enter the workings of a deep neural natural language processing model by proposing an original use of BERT embeddings. However, we have evaluated the quality of the predictions with the Jacquard index

3 Results

For task 1, the three bags of word embeddings reach roughly the same balanced accuracy on the test dataset (60%) which is twice as much as the majority classifier does. The BERT model reaches a balanced accuracy of 75% which is about 15 points greater than the bag words's performance (Figure 1). It means that even in a small document such as a tweet, the context of the word does matter a lot and that not taking into account the emoticons and hashtags is not at all prejudicial for the BERT model.

In Figure 2, We compare the recall of Bert and Count for each sentiment. 80% of the negative sentiments predicted by BERT were indeed tweets with a negative sentiment against 60% for Count. BERT widens the gap with Count on "negative" tweets. We can see that BERT predicts a negative sentiment in nearly 70% of the tweets with a negative sentiment against about 45% for Count. The performance gap is huge! We will therefore focus the qualitative evaluation on negative tweets that were badly "recalled" by Count but that were well recalled by BERT (cf Notebook output). We find that Count often predicts neutral sentiment instead of negative sentiment. The negative sentiment of most tweets is obvious to the human reader. First, key words or phrases like "rubbish", "late", "tired", "killing" etc. unambiguously convey negative sentiment. As the training base is relatively small (14000 tweets), it is likely that these negative words did not appear often enough to adjust the model to take them into account. Here, therefore, it is the pre-training of embedding on very large text corpora that gives Bert an advantage over Count. Secondly, forms of negation appear frequently in these negative tweets: "not happy", "no right", "i didn't even" etc. Since the negation term often precedes a neutral or positive term, the two terms then conflict in predicting the sentiment of the tweet. For example, in the tweet "This is not a happy tweet", the term "happy" trumped the term "no" because Count does not take into account the interdependence of the words (the context). Here, it is therefore the fact that Bert takes into account the context of the words that gives an advantage to Bert over Count.

For task 2, the Jacquard index is 67%. This means that on a set of 100 predicted and target support words, 70 were correctly predicted targets. This is a remarkable performance given the small size of the training data. We have created an evaluation function that decodes the predicted word vectors in order to qualitatively evaluate the prediction of the support words (see notebook output).

4 Discussion

Task 2 is very useful to understand the logic of BERT predictions in task 1. So these two tasks are related and that is why we wanted to do them together in this project. For example, we can use

the results of task 2 to understand why BERT is wrong in task 1. Indeed, knowing both the predicted sentiment and the predicted support words gives us an idea of the words that motivated the decision to predict. Due to time constraints, we were unable to link them in practice in this project. But task 2 provides an advanced qualitative evaluation method for any supervised text classification we should have to perform in the future.

5 Annexe

Figure 1: Comparison of balanced accuracies for each model

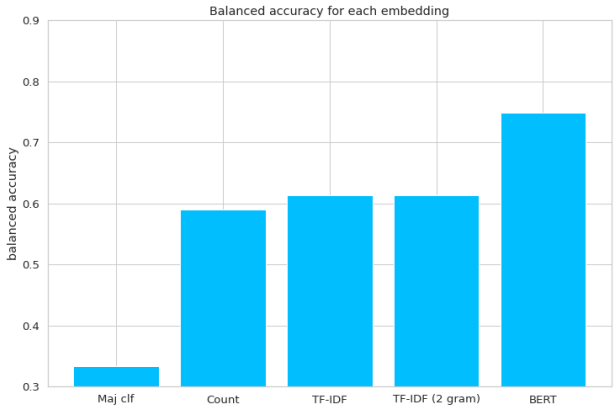
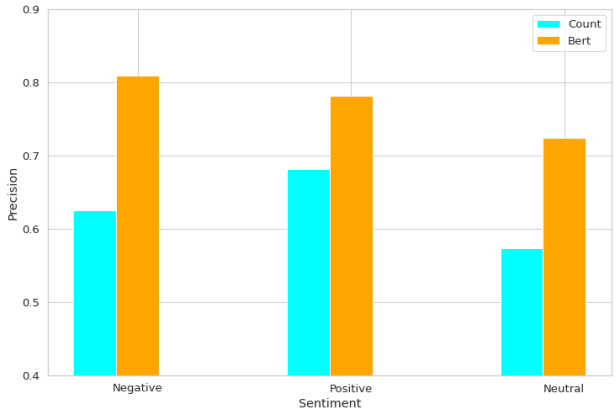


Figure 2: Comparison of Recalls for Bert and CountVectorizer



References

Kenton Lee Kristina Toutanova Jacob Devlin, Ming-Wei Chang. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*.