Aarsh Patel, Eleonora Quaranta

Twitter Sentiment Analysis

CS583 - Data Mining and Text Mining

Introduction

The aim of our project was the design of a pipeline for the sentiment classification of tweets on the two opponents of the 2012 United States presidential elections. We experimented with pretrained models to identify the best performing one on this specific task, and then used fine-tuning techniques to better refine its performance using the training data made available to us. We also tried to incorporate some Machine Learning algorithms to the pipeline, trying to correct possible biases of the sentiment classification model by predicting the final label from the different probabilities score for each class output by the fine-tuned model. However, this technique didn't improve the results on the evaluation data, which stayed nearly the same.

Techniques

Data preprocessing

The training data consists of two datasets of tweets, one about Obama consisting of 7198 records, and one about Romney consisting of 7200 records. Each record is characterized by the text of a tweet, the date and the time in which it was posted, and the corresponding sentiment class. The classes present in the training dataset were four, encoding different possible opinions on the candidate:

- -1: negative opinion
- 0: neutral opinion
- 1: positive opinion
- 2: mixed opinion

For this project we don't consider the mixed opinions, therefore we removed them from the training datasets, which led to a total of 5624 records for the Obama training set, and 5647 records for the Romney one. We also decided to drop the columns corresponding to the date and time of the tweets, since we think this information would not lead to a better sentiment classification.

As for what concerns the preprocessing of the text of the tweets, we performed some cleaning operations to remove irrelevant or misleading components, such as HTML tags, usernames and URLs. A popular preprocessing technique in natural language processing is the stopwords removal, which allows to reduce the dimensionality of data by discarding all those common words that don't have a major impact on the overall sense of a sentence. However, considering the specific task of interest, we didn't think that this operation would be beneficial for the performance of the model, since stopwords can play a crucial role in determining the overall sentiment of a sentence.

Pretrained Models for Sentiment Analysis

One of the most powerful tools for this type of task in Natural Language Processing are pretrained models, which are models trained on a very large corpora of text. Based on the task of interest, different models can be found and used to obtain better results compared to those that could be achieved when using a model trained from scratch on smaller data.

We explored the current models available for sentiment analysis, restricting our research to the field of tweet analysis because of the peculiarities of this type of text. The two main models that we identified and

tested are VADER (Valence Aware Dictionary sEntiment Reasoner)[1] and BERT (Bidirectional Encoder Representations from Transformers)[2].

- VADER: the VADER pretrained model[1] relies on a dictionary mapping lexical features to the sentiment scores indicating the polarity and the intensity of emotion expressed in the text, and on five heuristic rules to refine the sentiments identified. One of its advantages is its ability to process social media data, which has different characteristics in terms of syntax and semantics compared to traditional texts.
- BERT's main characteristic is its employment of bidirectional training, meaning that it processes text in both directions at the same time, allowing a better understanding of the context of each token of a sentence. The training operations are performed with the aid of a transformer with bidirectional self-attention. The training of BERT focuses on two tasks: Masked Language Model (MLM), which randomly masks one of the tokens of the input and tries to predict it based on the context, and Next Sentence Prediction (NSP), which consists in identifying how related two sentences are.

  One of the most popular models derived from BERT is RoBERTa (Robustly Optimized BERT Pretraining Approach)[3], developed by Facebook's AI division. The main differences from plain BERT consists of the more efficient training procedure characterizing RoBERTa, which allowed training on a larger dataset. The RoBERTa model lacks the NSP task and uses dynamically changing patterns for the masking. Furthermore, this extension leverages bigger batch sizes and longer sequences for the training, which benefits both the parallelization and the final accuracy of the model.

  BERTwee[4] is another development of the BERT base architecture, trained following the same procedure implemented by the developers of RoBERTa on a dataset of English tweets, thus making it particularly suitable for our task of interest.

Both VADER and BERT, including the RoBERTa variation, have been used as base architectures for more complex models, tailored on specific tasks. Among these models, we identified two extensions of BERT specifically implemented for tweet analysis:

- Twitter-roberta-base-sentiment-latest[2][5,6]: RoBERTa-based model trained on more than 124 million tweets and fine-tuned for the sentiment analysis task.
- bertweet-base-sentiment-analysis[3][7]: BERTweet-based model trained on approximately 40 thousand tweets.

Model Selection

At first we decided to test out three base pretrained models as we found them: VADER, twitter-roberta-base-sentiment-latest, and bertweet-base-sentiment-analysis, and compare their results on the two datasets joined together in order to select the best performing one and proceed to perform fine-tuning using the training data available. After this, we thought to leverage the fact that a model's output on the sentiment analysis on a tweet consisted of three probability values, one for each possible class. We thought that one possible way to improve the results could be to train a Machine Learning model, to which we feed a dataset containing three columns for the three different class probabilities and the target column with the true class as the training set. In this way, at test time, by feeding the model

[1] https://github.com/cjhutto/vaderSentiment
[2] https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment-latest
[3] https://huggingface.co/finiteautomata/bertweet-base-sentiment-analysis

simply the three probability values, we could get the adjusted class prediction hoping that the ML model would be able to correct the possible biases that a model could have, and thus correcting some of the mistakes made by it. We tested different algorithms for this purpose: SVM, Random Forest, KNN, and Logistic Regression, and performed k-fold cross-validation to tune the parameters to use for each model. In particular, the parameters that we considered for each model were:

- SVM: {C : (0.1, 1, 10, 100); kernel : (rbf, poly, linear, sigmoid); gamma : (scale, auto)}
- Random Forest: {n_estimators : (50, 100, 150); criterion : (entropy, gini); max_features : (None, 'sqrt')}
- KNN: {n_neighbors : (1, 3, 5, 7, 9); metric : (minkowski, euclidean, manhattan, cosine)}
- Logistic Regression: {penalty : (l1, l2, elasticnet, None), C : (0.1, 1, 10, 100), solver : (lbfgs, liblinear, newton-cg, newton-cholesky, sag, saga), max_iter : [500]}

Experiments Results

The first experiment that we performed was the comparison of the three base pretrain models listed above, using the whole training data as a test dataset:

MODEL	ACCURACY	PRECISION	RECALL	F1
VADER	0.3513	[0.7049 0.3324 0.7161]	[0.0354 0.9737 0.0824]	[0.0673 0.4957 0.1478]
twitter-roberta-base-sentiment-latest	0.5918	[0.6811 0.4731 0.6689]	[0.7222 0.6174 0.3279]	[0.7010 0.5360 0.4400]
bertweet-base-sentiment-analysis	0.6247	[0.6869 0.5132 0.6808]	[0.7621 0.5573 0.4716]	[0.722 0.5343 0.5563]

Seeing these results, we decided to proceed using the BERTweet-based model. At this point, we experimented with the fine-tuning process. Specifically, we tried to fine-tune two distinct models, one for the Obama tweets and one for the Romney tweets, and then tried again to fine-tune a single model joining the two datasets. For this section, we split the full datasets into a training, a validation and a test set, in order to be able to evaluate the performances; we used the train and validation sets for the fine-tuning process, and the test set for the final evaluation of each model. The fine-tuning process consisted of 2 epochs for the Obama model, 3 epochs for the Romney model, and 3 epochs for the unique model. The choice of the number of epochs for the fine-tuning were based on the need not to overfit the model. Here are the accuracies and losses of each of the models during the fine-tuning, computed on the evaluation set:

Epoch	Training Loss	Validation Loss	Accuracy
1	No log	0.748322	0.704000
2	0.765500	0.790275	0.716444

Obama model

Epoch	Training Loss	Validation Loss	Accuracy
1	No log	1.109888	0.496889
2	0.803200	1.178195	0.551111
3	0.502900	1.480066	0.600000

Romney model

Epoch	Training Loss	Validation Loss	Accuracy
1	0.858800	0.775103	0.660603
2	0.597200	0.893185	0.696983
3	0.376800	1.039635	0.691216

Unique model

Whereas these are the results of each model on the test set:

MODEL	ACCURACY	PRECISION	RECALL	F1
Obama	0.7288	[0.7209 0.6691 0.8082]	[0.7361 0.6828 0.7718]	[0.7284 0.6759 0.7896]
Romney	0.7177	[0.7451 0.6713 0.7013]	[0.8354 0.5229 0.7524]	[0.7877 0.5879 0.7259]
Unique	0.7024	[0.75 0.6242 0.7077]	[0.7866 0.5836 0.7051]	[0.7679 0.6032 0.7064]

We decided to move forward with the two separate models, since they seem to lead to better results overall, even if the difference is not that significant.

We used the outputs of the two separate models to evaluate the possibility of adding a ML algorithm to the pipeline, as explained above, but the results obtained with the different models did not lead to an improvement of the performance, but in most cases it made it worse. Therefore, we decided that our final sentiment classification pipeline should be composed of only the fine-tuned version of the bertweet-base-sentiment-analysis model, and its corresponding tokenizer.

For the final predictive model, we started again with two base bertweet-base-sentiment-analysis models, and then we fine-tuned each of them on the full training set for the corresponding politician. We then saved the final models in order to be able to load them at test time and run them on the new data.

Conclusions & Lessons Learned

In conclusion, with this project we aimed to develop a pipeline for sentiment classification on tweets about the 2012 U.S. presidential election candidates, Barack Obama and Mitt Romney. After experimenting with three different pretrained models, we found that the bertweet-base-sentiment-analysis model is the best performing one for the task of interest. After this, we used fine-tuning leveraging the separate datasets corresponding to the two candidates, which led to an even better performance. Despite our attempts to incorporate machine learning algorithms to correct possible biases introduced by the pretrained models, this approach did not improve the performance consistently. Therefore, our final sentiment classification pipeline relies exclusively on the fine-tuned version of the bertweet-base-sentiment-analysis model and on the corresponding tokenizer.

The project provided us with important insights on the challenges of sentiment analysis on political texts, highlighting the necessity for models to be able to correctly recognize opinions expressed on social media. The choice of using pretrained models significantly influenced the performance of our pipeline, but additional improvements were achieved through the use of fine-tuning, which proved to be crucial to our task. Even though the attempt to integrate machine learning for bias correction did not consistently improve results as we had hoped, it was still an interesting possibility to evaluate. Moreover, the decision to employ separate models for individual candidates highlighted the importance of having tailored models for each task.

Overall, this project allowed us to make important considerations on the sentiment analysis task, and notice how the different elements of a pipeline influence the final performance of the model.

References

[1] C. Hutto and E. Gilbert, “VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text”, ICWSM, vol. 8, no. 1, pp. 216-225, May 2014.

[2] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding", NAACL-HLT, 2018.

[3] Liu, Y., “RoBERTa: A Robustly Optimized BERT Pretraining Approach”, arXiv e-prints, 2019.

[4] Nguyen, D. Q., Vu, T., and Nguyen, A. T., “BERTweet: A pre-trained language model for English Tweets”, arXiv e-prints, 2020. doi:10.48550/arXiv.2005.10200.

[5] J. Camacho-Collados, K. Rezaee, T. Riahi, A. Ushio, D. Loureiro, D. Antypas, J. Boisson, L. Espinosa-Anke, F. Liu, E Martínez-Cámara, G. Medina, T. Buhrmann, L. Neves, F. Barbieri, "TweetNLP: Cutting-Edge Natural Language Processing for Social Media", Conference on Empirical Methods in Natural Language Processing, 2022.

[6] D. Loureiro, F. Barbieri, L. Neves, L. Espinosa-Anke, J. Camacho-Collados, "TimeLMs: Diachronic Language Models from Twitter" Annual Meeting of the Association for Computational Linguistics, 2022.

[7] J. M. Perez, M. Rajngewerc, J. C. Giudici, D. A. Furman, F. M. Luque, L. A. Alemany and M. V. Martinez, “pysentimiento: A Python Toolkit for Opinion Mining and Social NLP tasks”, 2021.