Twitter Sentiment Analysis with Various Model

Yu Xiao

Abstract

Sentiment analysis is a well-known and challenging task in NLP. We need to capture sentiment, semantic and sequential information simultaneously. In this project, I tried to experiment with different machine learning models with great performances on Twitter sentiment classification task and analysed their differences. I started with some common algorithms such as Support Vector Machine(SVM), Naive Bayes(NB) and Logistic Regression(LG), and then experimented with Neural Networks models such as Word2Vec, Convolutional Neural Networks(CNN), Long Short Term Memory(LSTM) / Recurrent Neural Network (RNN) and the combination of CNN and LSTM. Experiment results show that all Neural Networks models outperformed non-Neural Networks models. LSTM model achieved best classification result. Code is available at https://github.com/troyxiao/TwitterSentimentAnalysis.

1. Introduction

Sentiment analysis has a wide range of applications in real world. For example, it can be useful to monitor social media and extract general opinion, analyse survey responses and reviews, etc. Twitter is a great data source for sentiment analysis problem since tweets are short and have high emotional polarity. People usually use machine learning algorithms such as Logistic Regression(LG), Support Vector Machine(SVM) or Naive Bayes(NB) to study them. These algorithms typically take in texts as fixed-length vectors that represent the importance of words. TF-IDF vectorizer and Countvectorizer are two common vectorizers to extract features. In general, LR and SVM with linear kernel perform comparably in practice. SVM tries to find the widest possible separating margin, while LR optimizes the log likelihood function, with probabilities modeled by the sigmoid function [1]. Both LG and SVM are suitable for binary classification task. NB is another text classification algorithm based on Bayes' Theorem, that could determines the probability of different data attributes being associated with a certain class. Despite NB ignores the connectivity between words, it is still popular due to its simplicity. Compared to Neural Networks models, these three classifiers have some drawbacks. For instances, sequential information is ignored since the word order is lost. Neural Networks models can capture sequential information and semantic information to some degree. To be more specific, CNN model is able to learn the local features from words or phrases in different places of texts, while RNN model takes words in a sentence in a sequential order and is able to learn the long-term dependencies of texts rather than local features[2]. In this project, I experimented with Neural Networks models including Word2Vec, LSTM/RNN, CNN and the combination of CNN and LSTM. Comparisons among Neural Networks models and non-Neural Networks models are presented in the later section.

2. Related Work

Sentiment analysis is natural language process task with the goal to classify text to binary, or multiple sentiment classes. Early researches show that NB, SVM and LG are popular algorithms that are effective to such task. Compare to NB, the SVM method is statistical learning dependent hence it produces higher accuracy and precision when calculating polarity [3]. Many experiments on twitter sentiment analysis like [4] indicate that SVM and LG provided similar results. With the development of deep learning, word embeddings, CNNs and RNNs have been applied to text sentiment analysis and gotten remarkable results [2]. Experiments suggest that even a simple one-layer CNN could serve as a drop-in replacement for well-established baseline models, such as SVM or logistic regression [5]. Deep CNN models are expected to outperform non-Neural Networks models. LSTM, a part of the RNN family, with multiple variances such as Tree-structure LSTM and bidirectional LSTM, are good at capturing semantic relatedness [6]. The combination of CNN and LSTM has been extensively studied as well. The deep learning architecture of our model takes advantage of the encoded local features extracted from the CNN model and the long-term dependencies

captured by the RNN model [5]. The challenging part is to set up the architecture of model and choose proper parameters so that the model would yield the best classification result.

3. Data Analysis

3.1 Dataset

Sentiment140 is a labelled dataset that contains 1.6 million tweets collecting from Twitter API. There are 6 fields in the data file but only 2 fields are used in my projects. They are the polarity of tweets and the text of tweets. The dataset is balanced with 800,000 positive sentiment tweets and 800,000 negative sentiment tweets. The data file will be splitted to train and test sets.

3.2 Data Preprocessing

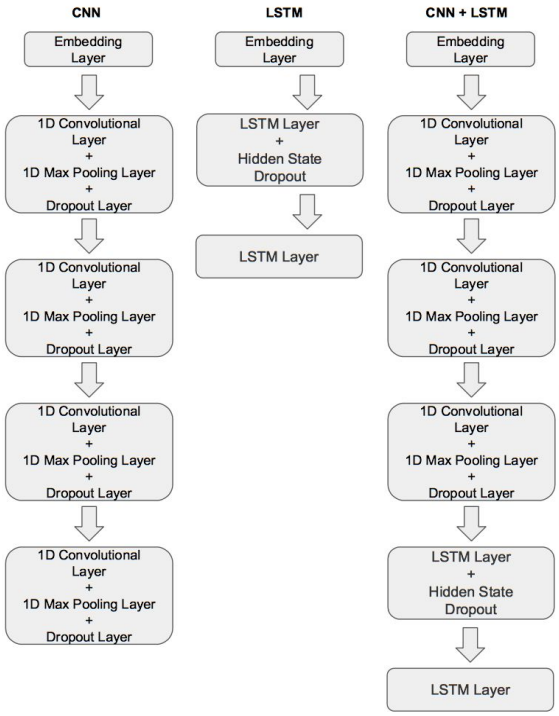
The first step of data processing is to convert all tweets into lower case. Then I remove all urls, hashtags, @username and multiple white spaces.

4. Approaches

For all non-Neural Networks models, I first tokenized and lemmatized the tweets. Then, I used TF-IDF vectorizer to convert tweets to feature vectors. Lastly, I splitted tweets into train and test sets and fed them into LG, NB and SVM models from sklearn library. I also tried to add Chi-square as dimensionality reduction for potential improvement.

For Neural Networks models, I tokenized all tweets and converted them to sequences with max length 20, then feed them into customized CNN, LSTM, CNN and LSTM models. The details of 3 models are as following:

- CNN model: one embedding layer followed by four convolutional layers. Each convolutional layer was following by a max pooling layer and a dropout layer.
- LSTM model: one embedding layer followed by two LSTM layers. The first LSTM layer has dropout units at each time step.
- CNN+LSTM model: one embedding layer followed by three convolutional layers and two LSTM layers. Same as previous models, each convolutional layer was following by a max pooling layer and a dropout layer. The first LSTM layer has dropout units at each time step.

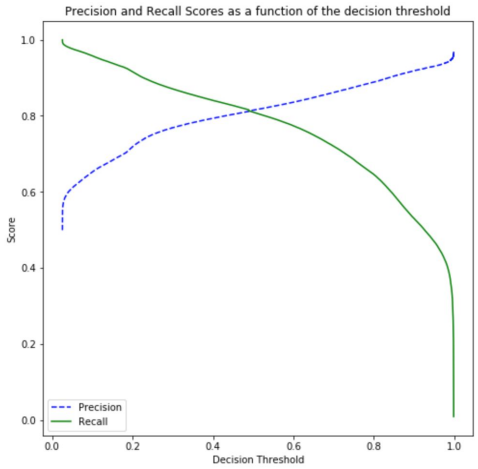


- Word2Vec: I first tokenized and lemmatized the tweets. Then, I used TF-IDF vectorizer to convert tweets to feature vectors. Lastly, I trained a word2vec model based on vectorized tweets and connected the model with fully connected layer.

5. Results

5.1 Evaluation Matrix

The evaluation matrix of models is weighted F1 score, the average of precision and recall. F1 score takes both false positives and false negatives into account and reflects more accurate classification ability than accuray, precision or recall. For Neural Networks models, I used precision and recall curve to find the best decision threshold that maximized both precision and recall, and then used such threshold to determine the weighted F1 score.



5.2 Evaluation Results

For non-Neural Networks models, the best F1 score is achieved by LG:

Model	With Tokenization & Lemmatization	Without Tokenization or Lemmatization
NB	0.77647	0.77650
LG	0.79042	0.79185
SVM	0.78098	0.78367

Experiment with dimensional reduction seems worsen the classification:

Model	Chi-square as dimensional reduction	Without Tokenization or Lemmatization
NB	0.71229	0.77650

All Neural Networks models achieved higher F1 scores than non-Neural Networks models. The best F1 score is achieved by the combination of CNN and LSTM:

Neural Nets Model	With Tokenization & Lemmatization
CNN	0.81422
LSTM	0.81643
CNN+LSTM	0.81714
Word2Vec	0.79557

6. Analysis

For non-Neural Networks models, it is expected that both LG and SVM would outperform NB since NB assumes the features are independent and basically just calculates the product of probabilities for each individual feature, but apparently local connectivities exist for tweets. Both SVM and LG are commonly used for binary classification task. SVM is good at using the kernel trick to solve the separation problem in a higher dimensional space. But in our case, tweets are pretty short and there is no need to project them to a very high dimensional space. Therefore, LG might yield a slightly better classification result than SVM. The other finding is adding Chi-square as the dimensional reduction worsens the classification result. This indicates that dimensional space is not a problem in our task and it serves as the other evidence that projecting vectors to a higher dimension is not needed.

All Neural Networks models outperform non-Neural Networks models. Comparing to non-Neural Networks models, Word2vec, an embedding of words into a low dimensional space, suggests that a dense embedding layer is a good first step. But among all Neural Networks models, Word2Vec is the shallowest networks and it yields the lowest F1 score. This indicates that deeper networks would work better. CNN and LSTM achieved similar F1 scores but LSTM did a little better. It appears that capturing long-term dependency is more important than capturing local repeating features. At last, as expected, the combination of CNN and LSTM outperforms CNN or LSTM alone since it takes advantage of both models. However, the challenging part is to find the right architecture of models. Adding too many CNN layers before LSTM layers would worsen the result since the sequential information would be lost with too many CNN layers, and in that case, LSTM layers would work similarly to fully connected layers. Overall, it is hard to determine the number of layers in models that would give best performance without experimenting. A deeper CNN and LSTM models are expected to yield better classification results.

6. Conclusion

In general, Neural Networks models have great performance on sentiment classification task. Comparing to traditional classifiers, Neural Networks models achieved a small improvement in F1 score in my experiments, but it is expected that Neural Networks models could yield a better result with carefully design and fine-tuning. Overall, all models experimented in the project did well in analysing sentiments polarity. Future works can be done on studying hybrid models that combine the strength of individual models.

REFERENCES

[1] Drakos, Georgios. "Support Vector Machine vs Logistic Regression – Towards Data Science." *Towards Data Science*, Towards Data Science, 12 Aug.2018,towardsdatascience.com/support-vector-machine-vs-logistic-regression-94cc2975433f.

[2] Wang,Xingyou, et al. "Combination of Convolutional and Recurrent Neural Network for Sentiment Analysis of Short Texts." The 26th International Conference on Computational Linguistics. 2016.

[3] Kaur, Wandeep and Vimala Balakrishnan. "Sentiment Analysis Technique : A Look into Support Vector Machine and Naive Bayes." 2016.

[4] Kursuncu, Ugur, et al. "Predictive Analysis on Twitter: Techniques and Applications." *Lecture Notes in Social Networks Emerging Research Challenges and Opportunities in Computational Social Network Analysis and Mining*, 2018, pp. 67–104., doi:10.1007/978-3-319-94105-9_4.

[5] Ye Zhang and Byron Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820.

[6] Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. *Improved semantic representations from tree-structured long short-term memory networks.* arXiv preprint arXiv:1503.00075.