A Comparison of Machine Learning Models for Binary Sentiment Classification

Joshua Mitchell

Abstract

Binary sentiment classification is process of accurately assigning a positive or negative sentiment label to a piece of natural language. Over the years, many natural language processing based models have been proposed to tackle this problem. In this paper, I train and evaluate the performance of three orthogonal machine learning models (Naive Bayes, Random Forest, and a deep learning based model called BERT) on the binary sentiment classification task. Experiments show that (1) right out of the box, the less complex methods perform better, and (2) the more complex models need a lot of engineering work to match the performance of simpler models (much less beat them).

1. Introduction

Sentiment Analysis (also known as Opinion Mining) has come a long way as a technology and a subfield of machine learning. Defined more formally, it is the problem of assigning a sentiment value to a natural language entity. Binary Sentiment Classification is the problem of assigning a positive (+) or negative (-) sentiment to a piece of natural language.

There is a long history of methods for solving this problem. Beginning attempts include using look-up tables and other knowledge-based techniques (such as counting the number of words in the document corresponding to a given sentiment label and weighting the document's overall sentiment based on the frequency of those words). More modern (but less cutting edge) techniques rely on traditional statistical and machine learning techniques such as Naive Bayes, Random Forests, and Support Vector Machines.

However, more recent developments in sentiment analysis rely on deep learning for astounding results in classification accuracy. While transfer learning (in this case, the process of using an initial deep learning model pre-trained on a large, standardized dataset and fine-tuning it on a new dataset) has become a standard problem-solving tool in computer vision, natural language processing is just beginning to utilize this concept (at least, in the form of deep learning) (Ruder, 2018), and is already seeing fantastic progress. Building on top of work on neural language models (Bengio et al., 2003) like word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), language-based deep learning models such as ELMO (Peters et al., 2018), BERT (Devlin et al., 2018), and XLNet (Yang et al., 2019) have started the transfer learning revolution in natural language processing.

In this paper, I evaluate the performance of a pre-trained BERT model (both baseline and fine-tuned), a Random Forest model, and a Naive Bayes model on three different datasets (Eight, 2019) (Bittlingmayer, 2019) (Hourrane, 2018).

1.1. Related Work

To give a more thorough treatment on work related to the binary sentiment classification problem, I will elaborate on two clever ideas whose emergence would revolutionize sentiment analysis (as well as a variety of other NLP tasks): word embeddings and neural networks.

Although the idea for neural networks was born in the 1950's (Rosenblatt, 1958), it wasn't until 2003 that an effective model for neural network-based language modelling was introduced (Bengio et al., 2003) (a large inspiration for existing models today).

Later, Google introduced word2vec (Mikolov et al., 2013), a method for representing words as distributed vectors in a substantially smaller vector space. These representations have been the backbone of a huge percentage of natural language processing systems. A similar story applies to GloVe (Pennington et al., 2014) (another common and often more effective technique for creating local word embeddings).

Meanwhile, in the deep learning field, there were three key ideas that needed to be introduced to give natural language processing its ImageNet moment: sequence to sequence models (2014), attention (2015), and memory (2015).

. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

In 2014, (Sutskever et al., 2014) introduced seq2seq learning: a general framework for mapping one sequence to another using neural networks. It does this using both an encoder to process a sentence token by token and compress it into a corresponding vector representation, and a decoder to predict the output token by token using previous token predictions.

Unfortunately, the biggest setback of seq2seq learning is the requirement that the entire source sequence be compressed into a vector of fixed length. Attention (Bahdanau et al., 2015) is an innovation on top of seq2seq learning that allows the decoder to look at weighted averages of the latent states of the source sequence (enabling decision making based on specific parts of the input).

During a similar time frame, progress and development of memory-based networks (Weston et al., 2015) exploded, resulting in deep learning models with robust state manipulation functionality.

Once these ideas were in place, they paved the way for the explosion of models and techniques for natural language processing and deep learning based transfer learning. In 2018, (Peters et al., 2018) introduced ELMO: a robust set of word embeddings learned from the intermediary representations of a deep, bi-directional LSTM language model. This raised the bar for State Of The Art (SOTA) on six different NLP tasks, and is largely heralded as the beginning of natural language processing's ImageNet era.

Soon after ELMO, a team at Google introduced BERT (Devlin et al., 2018) (Deep Bidirectional Transformers for Language Understanding): a technique for creating "deep bidirectional representations from unlabeled text by jointly conditioning on both the left and right context in all layers." This means that a BERT model can be used as a baseline model and fine-tuned for other tasks with just one additional output layer.

Finally, current SOTA models, such as XLNET (Yang et al., 2019), have been introduced that are essentially BERT models with slight augmentations to either the layers or the data for binary sentiment classification (Xie et al., 2019).

In other words, nowadays, solving natural language problems problems (like binary sentiment classification) looks a lot like solving computer vision problems: downloading a pre-trained model and attaching another layer.

1.2. Current SOTA Methods and Results

The current SOTA in binary sentiment classification (that I am aware of) is work by (Xie et al., 2019). Using a baseline BERT model, they propose to "substitute traditional noise injection methods with high quality data augmentation methods in order to improve consistency training." The name of their method is Unsupervised Data Augmentation or UDA.

Using a baseline BERT model and UDA, they achieved the above results on various datasets (see Table 1).

Table 1. Classification accuracies for BERT with Unsupervised Data Augmentation (UDA) on various data sets.

DATASET	NUMBER OF ROWS	ERROR %	ACCURACY %
YELP-2	560K	4.51	95.49
IMDB	25K	1.89	98.11
AMAZON-2	3.6M	2.63	97.37

2. Problem Description

In this paper, I evaluate the performance of a pre-trained BERT model (both baseline and fine-tuned for 20 and 100 epochs), a Random Forest model, and a Naive Bayes model on three different datasets (Eight, 2019) (Bittlingmayer, 2019) (Hourrane, 2018) for the purpose of binary sentiment classification. Below is a description of each method and dataset.

2.1. Methods

2.1.1. RANDOM FOREST

A random forest is essentially a weighted average of the results of multiple "random" decision trees. Each decision tree recursively splits data into categories based on features with the largest information gain.

In this case, the features are the word representations.

2.1.2. NAIVE BAYES

The Naive Bayes algorithm begins by assuming conditional independence among features to make calculations tractable. Then, it uses Bayes' rule to calculate the probability that a given sample x is of class C_k, for all classes k. Finally, for all samples x, assign x to be the class with the highest corresponding probability.

2.1.3. BERT

As mentioned earlier, BERT stands for Bidirectional Encoder Representations from Transformers, and is a technique for creating deep bidirectional representations from unlabeled text. It is essentially a Deep Neural Network with an embedding layer and a transformers layer (which are blocks of encoders and decoders).

A linear layer is added to the end of the model to complete the classifier.

2.2. Datasets

The data is composed of three datasets from Kaggle: IMDB Reviews, Airline Sentiment Tweets, and Amazon Reviews. What follows is a small description of each one:

2.2.1. IMDB

The Internet Movie Database (IMDB) is a site for, amongst other things, movie reviews. The IMDB dataset is a set of 25,000 highly polarized movie reviews for binary sentiment classification.

It is stored as a 23MB CSV file.

2.2.2. TWITTER

The "Twitter" dataset is a set of tweets scraped from Twitter concerning a group of major United States Airline companies during the month of February, 2015. Contributors were asked to first classify positive, negative, and neutral tweets, followed by categorizing negative reasons (such as "late flight" or "rude service").

The dataset is 8 MB in size and contains 15 features (I use the label and the text only).

2.2.3. AMAZON

The "Amazon" dataset is a set of 3.6 million reviews by Amazon customers. The dataset itself contains the customer reviews (input text) and star ratings (output labels).

It is stored as a 2 GB CSV file.

3. Results

Table 2 depicts the classification accuracy for each model.

Overall, it appears that Random Forest had most success for each dataset right out of the box, with the Twitter dataset seeing the highest accuracy of all datasets.

Naive Bayes saw a lukewarm amount of success, with accuracy ranging from 60% to 64%, barely above random chance.

Finally, the pre-trained BERT model saw no success at a baseline level (to be expected, with no training on the new datasets), lukewarm success with 20 epochs of fine-tuning, and moderate success after 100 epochs of fine-tuning (i.e. prediction accuracy ranging from 80% to 85%). This could indicate that more training is needed (since each dataset only went through 100 epochs of training). It could also indicate that more data is needed (since, in general, more data let to higher accuracy). I hypothesize that, since BERT was trained on all of Wikipedia, Amazon reviews would be more similar to what BERT was trained on.

Table 2. Classification accuracies for Random Forest, Naive Bayes, and BERT (untuned, 20 epochs fine-tuning, 100 epochs fine-tuning) on various data sets.

DATASET	RF	NB	BERT-0	BERT-20	BERT-100
TWITTER	0.897	0.64	0.402	0.565	0.798
IMDB	0.841	0.63	0.4834	0.659	0.828
AMAZON	0.853	0.6	0.4862	0.673	0.851

4. Conclusion

I evaluated the performance of a pre-trained BERT model (both baseline and fine-tuned for 20 and 100 epochs), a Random Forest model, and a Naive Bayes model on three different datasets with the intent of solving the binary sentiment classification. Some key contributions of the project include:

- illustrating that the Random Forest algorithm is a reasonably fast and efficient method to get good results
- showing that Naive Bayes is good for getting a really quick result
- showing that really complicated models like BERT require more time and expertise to get good results (even though my results aren't SOTA)

Include a brief summary of the main contributions of the project and lessons you learned from the project

4.1. Lessons Learned

There are two lessons I learned during this experiment: make sure to write efficient and optimized code that examines and iterates over the data, and a lot of success in creating accurate models comes from engineering.

4.1.1. WRITE EFFICIENT CODE

When it comes to examining millions of rows worth of data, a general rule is that a standard for loop is not going to run within a reasonable time frame. There are built in methods for various libraries that are optimized for this, and they should be used (such as df.apply, cython, vectorization, etc).

4.1.2. WRANGLE DATA PROPERLY

It turns out that the models are only tools, and that simply feeding the data as-is into a model will usually exhibit sub-par results. There are lots of considerations to make before the model training ever begins (such as: which data points should be thrown out (if any), are there some points that are mislabeled, how much data should be devoted to

training, testing, and validation, etc). For non-deep neural network based models, there is also a large degree of feature engineering that must take place.

4.2. Future Work

There are a lot of avenues that this project could take. To list a few possibilities:

- Exploring the performance of other traditional machine learning models (Support Vector Machines, Logistic Regression, etc)
- Try different natural language processing specific data wrangling schemes (different sets of stop words, different stemming and lemmatization schemes, etc)
- Try an even more recent Deep Neural Network based language model (like XLNet)
- Fine-tune the existing BERT model even more (going as far as I can while looking for signs of over-fitting)
- Modifying the BERT model to create something beyond semantically different from baseline

To expand on the last bullet point, there is a lot of non-novel variations I could experiment with, such as adding or removing layers, tweaking the types of layers (e.g. substituting different encoders or decoders), or rearranging layers.

As far as more novel contributions, that might involve creating a new layer. I don't have any ideas for that yet, but if I do, that will probably be the path I pursue.

References

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.0473.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3, March 2003. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=944919.944966.

Bittlingmayer, A. Amazon reviews for sentiment analysis. =https://www.kaggle.com/bittlingmayer/amazonreviews, 2019.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Eight, F. Twitter us airline sentiment. =https://www.kaggle.com/crowdflower/twitter-airline-sentiment/, 2019.

Hourrane, O. Imdb movie reviews for sentiment analysis. =https://www.kaggle.com/oumaimahourrane/imdb-reviews, 2018.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pp. 3111–3119, USA, 2013. Curran Associates Inc. URL http://dl.acm.org/citation.cfm?id=2999792.2999959.

Pennington, J., Socher, R., and Manning, C. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://www.aclweb.org/anthology/D14-1162.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep contextualized word representations. *CoRR*, abs/1802.05365, 2018. URL http://arxiv.org/abs/1802.05365.

Rosenblatt, F. F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65 6:386–408, 1958.

Ruder, S. Nlp's imagenet moment has arrived. 2018.

Sutskever, I., Vinyals, O., and Le, Q. V. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014. URL http://arxiv.org/abs/1409.3215.

Weston, J., Chopra, S., and Bordes, A. Memory networks. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1410.3916.

Xie, Q., Dai, Z., Hovy, E. H., Luong, M., and Le, Q. V. Unsupervised data augmentation. *CoRR*, abs/1904.12848, 2019. URL http://arxiv.org/abs/1904.12848.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J. G., Salakhutdinov, R., and Le, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237, 2019. URL http://arxiv.org/abs/1906.08237.