A Novel Ensemble NLP Sentiment Analysis Exploration

Yao Chen, Zixi Wang, Kevin Xuan

yaoc16@berkeley.edu
zwang2020@berkeley.edu
kevinxuan@berkeley.edu
School of Information, Univeristy of California, Berkeley

Abstract

Over the past decade, Sentiment Analysis has been getting extremely popular with the arrival of deep learning neural networks and has always been one of the frequent tasks to do in Natural Language Processing (NLP). Recently, the idea of ensemble method and multi-modal learning have also been widely used in different applications, and in this paper we also exploit model ensemble to derive the emotion that is conveyed in the text of a review. To extrapolate the sentiment of a paragraph of text, we start with a baseline model and then construct three separate models which individually focus on the following: polarity, stress, and sarcasm. We believe that these concepts have high correlation to whether the text is conveying a positive or a negative sentiment, and learning from a combination of the outputs of each model will allow us to earn higher performance in sentiment analysis.

Introduction

Sentiment analysis has been one of the most interesting topics in the domain of natural language processing. It is a study on extracting information from a given paragraph of text and determining whether the paragraph conveys a positive or negative sentiment. With the result, companies are able to monitor the feedback of its brand and products based on user reviews and quickly address those issues within short amount of time.

This machine learning task is usually a binary classification problem which aims at detecting whether a text has a positive or negative sentiment. However, language is extremely complicated and can be misleading, especially with words that have multiple interpretations. Language styles that stem from various environments such as Twitter [5] and Reddit [1] can be very unique and it is more difficult to interpret their text sentiment because of a mixture of expression involved. Sarcasm can involve positive wording but mean ironically. Stressful words can lead to more negative meaning than they appear. Polarity is a measure of the overall combination of the positive and negative emotions in a sentence within a context.

A mixture of these sentiment can confuse a model that learns a single sentiment. Using a single corpus may or may not include other confusing sentiments. An ensemble model that learns different sentiments from different corpus helps to identify confusing sentiment and thus improve model performance in detecting the primary sentiment which is defined by either positive and negative.

Related Work

In the past decade, there has been uncountable number of papers focusing on using deep learning neural networks in sentiment analysis. However, researchers try to solve the problem with different approaches. Starting with word embedding, that is to convert a paragraph of texts to vectors, researchers use a popular technique Word2Vec [9]. In terms of algorithm selection for their model predictions, most researcher use either Gated Recurrent Unit (GRU) [4] or Long-Short-Term Memory (LSTM) [6] as the base for their models which uses word embedding as input and output the polarity of the text.

In 2018, when Google introduced the concept of transformers [13] and Devlin introduced BERT [3], numerous researchers such as [2] began to use BERT to conduct word embedding on paragraph of texts and BERT model to make predictions on the embedding. Since BERT earns high performance in sentimental analysis, some researchers began to look into ensemble method with BERT. For instance, Tongwen [7] applied ensemble method to BERT on multiple NLP tasks. Hence, in our paper we also build an ensemble model via BERT embedding.

To create an ensemble model different from existing research, we look into concepts that are highly correlated to sentiment, and we find that sarcasm, stress, and polarity tend to have high correlation to predicting whether a post contains positive or negative contexts.

1. **Sarcasm** - using words that mean the opposite to what a person really want to express, and it usually involves mocking someone or something. Khodak [8] created a dataset, built a machine learning model to detect whether the text contains sarcasm,

and compared the performance of the model with that of a human being.

2. **Stress** - physically and mentally feeling of anxiety, anger, and frustration. Turcan [12] initially constructed various machine learning algorithm based models such as SVM and decision tree but then used neural network such as CNN and RNN to infer the sentiment of the text. However, she used BERT as the ultimate model to predict whether there is stress inside a paragraph of text.
3. **Polarity** - positive or negative sentiment. Elkouri constructed Naive Bayes, SVM, and Logistic Regression models on the Yelp Review dataset to detect whether the text contains positive or negative sentiment.

By ensemble the three concepts above, we provide additional support to a baseline model that is trained to predict positive/negative sentiment.

Methods

In this section, we explain how we create the final ensemble model consisting **Baseline**, **Sarcasm**, **Stress**, and **Polarity**.

Our objective is to train multiple models independently first and then combine them to come up with the ensemble model so that it is able to make predictions on multiple sentiments of a text. Before training all the sub-models separately at the start, we notice that all our models utilize the same text input. Hence, we create a data preprocessing pipeline that handles all texts input used for model or sub-model training.

By looking into the data, we notice that hundreds of reviews have no text descriptions, so for our training and evaluation purpose we remove all those data points. We then begin to examine data points with short amount of texts. We observe that reviews that have one or few words can also reflect a negative sentiment. For instance, *'Hated'* or *'This movie is a waste of time'* shows negative sentiment toward a movie, even though the length of the text is small. Hence, we decide to keep these data. Since most of sub-models' data have an average word length of 50, we exploit BERT tokenization with size of 50.

Next, we start our model training process. For all these sub-models, we use similar neural network architecture where we first feed BERT embedding into the model. For baseline model, sarcasm model, and polarity model, we apply BERT model on the embedding and extract the pooled token output from it. For stress, we also apply BERT model on the embedding, but different from others we obtain the CLS token output and take an average. Afterwards, we feed the output with a fully connected layer with dimension of 100 followed by a ReLU activation function and a 30% dropout.

With model training, the classic ensemble methods make prediction out of an averaged prediction from candidate models. Our proposed ensemble method encourages the model to learn from embedding generated from

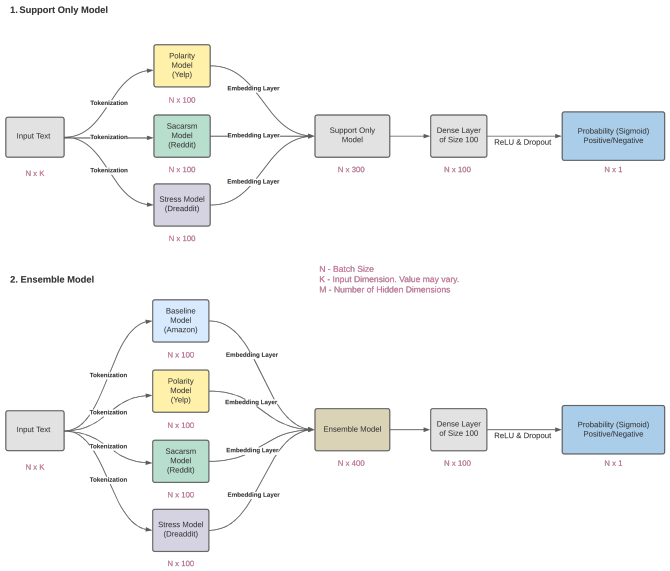

Figure 1: Flow Chart of Implementation of Ensemble Model

each sentiment model. For each sentiment model, we train it on a different corpus that is characterized by a particular kind of sentiment. After all 4 models have completed training, we store the weights of all the models up to the embedding layer vectors of size 100 prior to the final sigmoid layer. We then create a neural network architecture concatenating the 4 vectors of size 100, add a fully connected layer of size 100 followed by an activation function and dropout, and add another sigmoid layer transforming the vector to the final single output.

During the training process, weights of the 4 models – discussed below – are frozen in order to keep the learned sentiment of each sub-model on its corresponding concept, so we are training the weights only on the fully connected layer. We have completed our model training process and can then make predictions.

Baseline Model

To train the baseline model, we apply pretrained BERT tokenizer embedding $BERT - base - cased$ on the training data with the word size of 50. We then use a BERT model that reads in the three inputs of the BERT embeddings and outputs a pooled token vector. With the token, we add a fully connected layer of size 100 followed by a dropout of 0.3. Lastly, we add a sigmoid function the layer to obtain the final single output. For the optimizer, we apply Adam with a learning rate of 0.00005. With this architecture, we then put the data for training and evaluate our model on the test set.

Stress Model

Having too much stress leads to expression of complicated emotions, and therefore detecting stress helps de-

Figure 2: Entire Model Architecture of Ensemble Model on Sentiment Analysis

termine whether sentiment of a text is positive or negative. Stress model does the job by detecting whether a text contains stress. We trained the model with an annotated Dreaddit [12] dataset containing thousands of lengthy texts identified either as contexts that contain stress or that without. Similar to baseline model, we utilize BERT tokenizer to embed the training data. However, different from it, we use $BERT-large-uncased$ in the stress model, as we believe the case of a word might convey totally opposite interpretations. For the model architecture, we also utilize a BERT model that reads the BERT tokenizer output. Instead of obtaining the pooled token, we extract the three classification outputs and apply an average on them. Then similarly we add a fully connected layer with a dropout and a sigmoid function to derive the final output. In the end, we train our model with a total of 108,387,273 parameters and evaluate the performance on the test set.

Sarcasm Model

Sarcasm as one of the confusing sentiments, is learned from a BERT model that is trained on a dataset of 1.3 million Reddit comments. As a popular online forum where drastic debates often take place, Reddit involves an abundance of ironic comments. To obtain a better learning process, we use a balanced dataset (class ratio 1:1) to train the sarcasm model. For the model architecture, optimizer, and loss function, we utilize the same hyper parameters as those of baseline model. The sarcasm model is able to achieve an accuracy of 0.7704.

Polarity Model

Polarity takes into account the amount of positive or negative terms that appear in a given sentence. It often refers to the strength of an opinion which is widely used in classifying customer reviews. We choose the Yelp reviews dataset confined within similar language domain and positive/negative labels with a refinement of the sentiment analysis tasks. The polarity model is trained from 560,000 Yelp reviews using BERT model. Similarly, the hyper parameters such as optimizer and loss function are consistent with baseline model. The polarity model test accuracy score is able to reach 0.8450.

Ensemble Model

The goal of the proposed ensemble model is to leverage additional three sentiments to help the baseline model identify whether a text has a positive or negative meaning. Each sub-model uses its own tokenizer to process input text. The ensemble model concatenates the output of last second layer from each model to create an ensemble embedding layer. During the training, weights of ensemble layers are frozen while followed dense layers are allowed to update weights based on loss it sees. The final output is calculated by a sigmoid activation to predict whether a text means positive or negative.

Datasets

Amazon Review

Product rating is one of the frequent applications for Sentiment analysis. Hence, we decide to use the dataset of one of the largest E-commerce companies Amazon [10]. We choose two small subsets in the 2018 Amazon Review Data: Movies and TV (3,410,019 reviews) and Electronics (6,739,590 reviews). We believe these reviews tend to deliver positive and negative perspectives on products (either liking them or not) and reflect in a 5-score rating. The dataset contains metadata such as the time the review was written, the reviewer ID, review text, overall rating, etc. For our training and evaluation purpose, we are only interested in the two columns: review text and overall rating of products where we remove reviews with 3 - 4 ratings and categorize 1-2 ratings as negative (bad) reviews and 5 as positive (good) reviews. Based on the distribution of the ratings label, we notice a class imbalance issue in the dataset skewed towards 5-star positive rating. Therefore, we undersample some of the positive labels to alleviate the class imbalance impact.

Yelp Review - Polarity

The Yelp Reviews Polarity dataset [14] consists of highly popular 1-4 stars reviews from Yelp with positive (3 and 4 stars) and negative (1 and 2 stars) polarity. For each polarity, there is 280,000 training samples and 19,000 testing samples. This is a dataset for binary sentiment classification. In total, there are 560,000 training samples and 38,000 testing samples.

Reddit Dataset - Sarcasm

This dataset [1] is collected from the Internet commentary website Reddit, which includes 1.3 million sarcastic comments. The labels are created from ”/s” (sarcasm) tag in each comment. This tag is used by Redditors to indicate that the comment should not be taken seriously, which works as a reliable label to flag sarcasm. The true class ratio is 1 to 100 but there is also a balanced version. To assign equal weights to both classes, we use the balanced dataset for training the model.

Dreaddit Dataset - Stress

Dreaddit [12] is a dataset on stress anaylsis on social media from Subreddits, topic specified communities from Reddit. It consists of 190k posts from 5 different categories of Reddit community, and from these posts approximately 4,000 texts are labeled. The dataset contains columns like the category the text belongs to, Similarly to what we have done to Amazon Review dataset, we are only interested in extracting the column containing text of posts – with an average length of 420 words – and the column labeled whether the text contains stress or not.

Twitter

Twitter, one of the leading social media service companies, contains plenty of posts with strong expression of emotions. Hence, we download a class balanced Twitter dataset named Sentiment 140 [5] containing a total of 1.6 million posts, in which 8 million of the posts contain positive emotions while the other 8 million contain negative emotions. Similarly to some of previous datasets, this dataset also contains metadata such as user id writing the post and the date which the post is created. We extract the columns which contain the text of the post as our input to train a model and the column which includes the labels of whether the post contains positive or negative expression.

Results & Discussion

Performance Metrics

Our research problem is a binary classification problem, and so forth we utilize the frequently used binary classification metrics for our model evaluation. We notice that these datasets are all balanced, having a 1 to 1 ratio for positive and negative comments. For this situation, we believe accuracy is the most effective metric to measure the model performance because with the number we can clearly know how many posts the model predict correct or wrong. We have also considered other metrics like AUC, Precision, Recall, and F1-Score, but they are not as straightforward as accuracy.

Support-Only Model

With three trained sub-models, each specializing in detecting a different sentiment, it is interesting to gauge how being positive or negative is correlated to other sentiments. Therefore we run an experiment of using a support only ensemble. It uses concatenated embedding from three sub-models, excluding the baseline embedding. This support-only ensemble is able to achieve an accuracy of 0.8269 out of a mixture of stress, sarcasm and polarity. These sentiments are decently correlated to being positive or negative and shows sign that ensemble model may work.

Implementation Details

In the aspect of hardware, we create, run, and test our entire code on Google Colab. In terms of software, we mainly use transformers and TensorFlow package for our model training and evaluation process.

Method	Amazon	Twitter
Baseline	**0.9490 (0.1586)**	**0.9161 (0.2080)**
Ensemble	0.9416 (0.1652)	0.8960 (0.2578)

Table 1: **Accuracy** (**Loss**) of Baseline Model vs. Ensemble Model for Sentimental Analysis on Amazon Review Movies & TV and Electronics dataset and on Twitter Posts dataset

For all the datasets involved in our sub-model training, we focus only on two columns: an input which is a text description and a binary output in which 0 indicates the text contains negative sentiment and 1 indicates positive. We then utilize the conventional method train validation test to split the data with corresponding 60-20-20 ratio. For baseline model, we train a batch size of 64 for 5 epochs with Adam Optimizer of learning rate 0.00005. For the ensemble model, we train a batch size of 128 for 5 epochs with same optimizer and learning rate as of those of baseline. For both of these models, we use binary cross-entropy as our lost function to back-propagate our weights of the network.

Results

In this section, we evaluate our models on Amazon Review and Twitter posts, and we compare the performance of baseline model and that of ensemble model. From the model results, we see that the performance of the two models are close to each other. On the Amazon dataset, baseline model earns an accuracy score of 94.90% while ensemble model earns an accuracy of 94.16%/. For Twitter, baseline model earns 91.61% and ensemble model earns 89.60%.

Error Analysis

To evaluate the effectiveness of the proposed ensemble model, we look at the cases that are misclassified by the baseline model, and then compare their predicted probability with ones generated from the ensemble model. Specifically, we perform a detailed analysis for the false positives and false negatives. Overall, we find that the ensemble model tends to predict a lower predicted probability than the baseline as shown in Figure 3a. Thus, it is with our expectation that ensemble can help correct some false positives from baseline model predictions. From the Figure 3b, we see that ensemble tend to lower the predicted probabilities as a correction from additional sentiments. On the test set, the ensemble model is able to correct 651 cases misclassified by the baseline model but also introduce 727 erroneous predictions. Therefore, ensemble model is performing slightly worse than the baseline model. However, it is still interesting to see how the ensemble makes a correct prediction using additional sentimental information from texts.

The proposed ensemble is expected to use additional sentiments to correct baseline model. The below exam-

ple is where sarcasm is well detected by the ensemble model.The author mocks the maker of the movie by writing an official, short but insulting message.

Dear Makers Of Date Movie,
You Suck.
Kind Regards, Joey Bananas
"The Movie Monkey"

Baseline model predicts it as positive with a probability of 0.531388 but the ensemble corrects the prediction by outputting a probability of 0.474716, thus classifying it as negative. Yet another type of sarcasm is seen in the test samples as shown below. This review gives 5 stars as rating but clearly it shows no sign of anything positive from the text. This is an example of text confusing the model in a sarcastic way. But note that the author uses sarcasm in assigning the label but not in the language used. Therefore, it should be expected our ensemble model cannot learn this sophistication because it makes decision based only on the text.

Poor acting, tired storyline.

Stress is most obvious in text where emotional words are present to indicate how people feel at the moment. The example below contains a strong sense of being desperate and helpless. The ensemble correctly recognizes the stress that is relieved and uses it to predict a true positive, instead of relying on the many negative words to make predictions as the baseline does.

I don't know what to say. It works, which is in contrast to the customer service from DirecTV. The sales person said I didn't need a separate tuner to receive over the air broadcast including all the subchannels, which was wrong....

The polarity model works mostly for providing additional contexts of language style, as the learning task is similar to that of the positive or negative sentiment detection but with a different dataset. The contrast it creates between the baseline and ensemble model is less obvious compared to that of stress and sarcasm models.

We also identify some issues within the data. One issue is that Amazon Review dataset has some texts that are mislabeled. For example,

Love it, comfortable and has nice inside to protect my equipment.

Just what I need for my tablet; easy to use. A reliable Samsung product; my second purchase within a month (one for my smartphone as well). You can rely on Samsung.

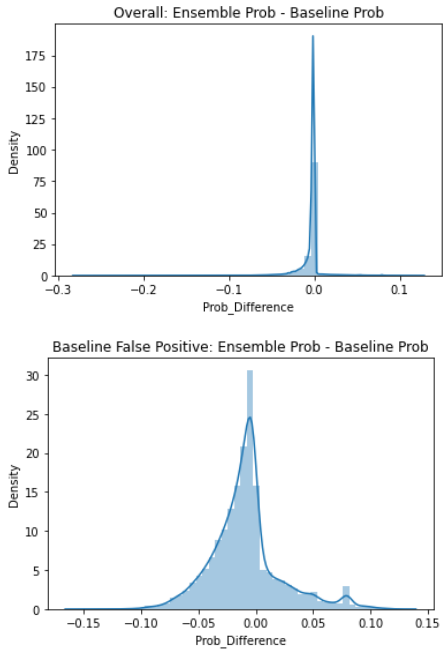

Figure 3:
3a: Probability Difference between Baseline and Ensemble
3b: Probability Difference between Baseline and Ensemble for Data Points Predicted False Positives

both show clearly positive sentiment but are labeled as negative. This results in higher false positives in our evaluation.

Another interesting phenomenon is that the baseline model seems capable of identifying the overall attitude of reviews while it cannot further identify if authors approve the products per se. There are examples that authors think a product is good but for some additional reasons they are labeled as negative sentiment.

It's looks good but the price dam if it was at 227.99 or 225.99 I would of purchase it

They work as advertised. This product didn't shake my world, but I'm 68, everything about me shakes. So if you are looking for a life altering experience, this isn't it.

Above are examples where authors are not satisfied because products are not perfect and only meets basic expectations. In those cases, review texts demonstrate some negative sentiments but labels are still considered positive because they are supposed to focus on the products. Such combination of text and label further complicates the learning tasks for the models.

Challenges

A good model – especially a neural network model– requires large quantity of data. Stress model, one of the

Method	Data	Accuracy
Baseline	Amazon	**0.9490**
Sarcasm	Reddit	0.7704
Stress	Dreaddit	0.6839
Polarity	Yelp	0.8450

Table 2: Sub-model Accuracy on Test dataset

sub-models, however has only a few thousands of data points to train on. Due to the small amount of training data, the model is not able to fully learn the characteristics of the data, and thus not able to perform well on the test set. As a result, we are only able to train the model to an accuracy of 68%.

Not only is there smaller size of data, there are other challenges:

- **Poor Transfer Learning** A good model works well only if the training data and test data are similar or have similar distribution. However for our sub-models, each model is trained on a different dataset focusing in different categories. In addition, the test set we are evaluating our models on is about another area of interests. For instance, the stress model is trained on Reddit dataset which involves discussion on a matter while our test set Amazon Reviews pivots on movies and electronics and Twitter specify on personal life posts. Consequently, when we apply transfer learning of a model being trained on a dataset that does not have strong correlation with the test set, we do not get good results.
- **Lower Performance on Sub-models** From the result section, we can see that the baseline model is able to achieve extremely high performance on the Amazon Review dataset, with an accuracy of 90+%. However, when we look into our sub-models, we see that the accuracy we get from the predictions [table 2] are much lower than those from baseline. Due to this situation, the sub-models induce their own bias to the ensemble model. There exist many cases where one sub-model performs really well but because others perform poorly. The poor results the good ones. As a result, the ensemble model consisting these models has even larger bias, which results in poorer performance for ensemble model.

Future Work

Due to time and capacity constraints, we only apply the ensemble model on Amazon Review and Twitter dataset which may not fully reflect other sentiments identified in the sub-models. The BERT baseline model is good enough to detect normal positive and negative sentiments. Therefore, in order to leverage the embedding layers generated from sub-models, we expect to see a better model performance when applying the ensemble model on a dataset with more sarcasm and stress contexts, such as Reddit crypto dataset with extreme emotions and phrases.

In addition, we initially hope to add a Convolutional Neural Network (CNN) [11] layer after concatenation as one of the potential processing step. If time permits, we want to explore whether the CNN layer will help boost the performance of our ensemble model.

Conclusion

In this paper, we construct an ensemble model – consisting of four sub-models focusing on specific areas – to improve sentimental analysis. All the data are preprocessed via BERT embedding, and all sub-models and the final ensemble model have a similar deep learning architecture. For the results, we compare our model between the baseline model and the ensemble model. Both models perform extremely well on the test datasets: Amazon Review and Twitter. Although we see poorer performance for the ensemble model on both datasets, we are able to find examples that a ensemble is able to predict better than the baseline model.

References

[1] 2017. A Large Self-Annotated Corpus for Sarcasm.

[2] Chiorrini, A.; Diamantini, C.; Mircoli, A.; and Potena, D. 2021. Emotion and sentiment analysis of tweets using BERT.

[3] Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186. Minneapolis, Minnesota: Association for Computational Linguistics.

[4] Dey, R.; and Salem, F. M. 2017. Gate-variants of gated recurrent unit (GRU) neural networks. *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*.

[5] Go, A.; Bhayani, R.; and Huang, L. 2009. Twitter Sentiment Classification using Distant Supervision.

[6] Hochreiter, S.; and Schmidhuber, J. 1997. Long Short-term Memory. *Neural computation*, 9: 1735–80.

[7] Huang, T.; She, Q.; and Zhang, J. 2020. BoostingBERT: Integrating Multi-Class Boosting into BERT for NLP Tasks. *CoRR*, abs/2009.05959.

[8] Khodak, M.; Saunshi, N.; and Vodrahalli, K. 2017. A Large Self-Annotated Corpus for Sarcasm. *CoRR*, abs/1704.05579.

[9] Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3781.

[10] Ni, J.; Li, J.; and McAuley, J. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. 188–197.

[11] O'Shea, K.; and Nash, R. 2015. An Introduction to Convolutional Neural Networks. *CoRR*, abs/1511.08458.

[12] Turcan, E.; and McKeown, K. 2019. Dreaddit: A Reddit Dataset for Stress Analysis in Social Media. 97–107.

[13] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All You Need.

[14] Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-Level Convolutional Networks for Text Classification . *arXiv:1509.01626 [cs]*.