IMDb Movie Reviews for Sentiment Classification

Pranoti Dhamal, Tianyi Zhang

December 2021

Abstract

Customers reviews for movies are an essential feature to know the quality of a movie and the preferences of viewers. This project is aimed at classifying a movie review to one of the positive or negative categories. Models are trained and tested on IMDB movie reviews. We experimented with 1 sparse doc representation and 2 dense doc representations and prove the efficiency of dense document embeddings. We employed 4 different traditional machine learning approaches and 2 deep-learning models. We improved the accuracy scores for about 10 percent to above 95% accuracy rate comparing with our logistic regression baseline model. We find the goodness of embeddings, model architectures, number of parameters in a model as well as transfer learning can all influence the performance of a task.

1 Introduction

It's essential for the movie industry, e.g. Netflix, Pixar, to know the viewers' likes and dislikes in order to continuously generate profits. Knowing customers' preferences not only gives inspiration for producing popular movies, but also improves the recommendations of potential interested movies to the target viewers.

One direct way of investigating peoples' favorite movies is learning from their reviews, seeing whether they have a positive or negative emotion about that movie. This is very meaningful but inefficient by using human brute force. And we are interested in employing NLP knowledge to solve this task and help the movie industry to generate more profits.

Identifying a positive, negative or neutral attitude of a certain piece of text (Sentiment Analysis) is a common and well-developed classification task in the NLP area. It has many related annotated datasets, e.g. Twitter, Amazon product, IMDb, to work on and various classification models (traditional machine learning or modern deep learning) can help. With this knowledge, we are convinced that our task of understanding viewers' emotional attitude to a certain movie according to their reviews is solvable by employing NLP methods and machine learning models.

The main goals of this project are:

- Search for an efficient document representation for classification models.
- Build a machine learning baseline classification model, e.g. Logistic Regression.
- Experiment with variations of machine learning models, e.g. Naive Bayes, Neural Nets, ensemble models to improve accuracy score.
- Further explore on modern approaches, e.g. GPT-3, for sentiment classification.

2 Related Work

The most related work to our project is from Andrew L. Maas, et. al, (2011)[1]. They first proposed the IMDb movie reviews dataset. In their work, they discussed several possible document vectors and machine learning models to classify the reviews to positive and negative. They showed that Bag of Words exceeded LDA, probability models which encoding some semantic meanings of words + Bag of Words (their approach) worked best. Duyu Tang, et. al, (2014)[2] also presented the similar idea of generating sentiment word embeddings to facilitate sentiment classification.

Later, with the emergence of deep learning models in NLP area, e.g. LSTM, Transformer, many researchers focused on deep NN to produce contextualized sentiment specific embeddings that can benefit sentiment analysis. For example, Usman Naseem, et. al, (2020)[3], proposed to apply Transformers to generate sentiment word or doc embeddings targeting at classifying tweets. Dat Quoc Nguyen, et. al, (2020)[4] proposed a BERTweet model specifically for English tweets.

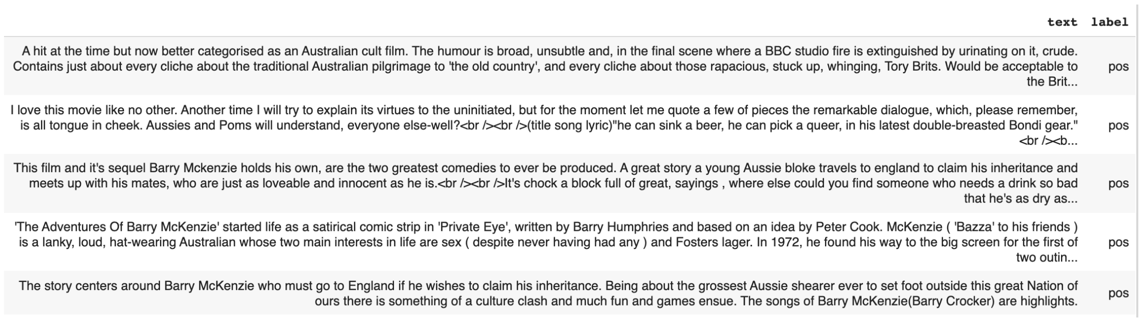
In our approach, we experimented with both traditional doc vectors, more specifically, Andrew L. Maas's emotional values for each word + word counts (a similar but simple approach of their document vector model) and current contextualized doc embeddings, specifically, [CLS] and average embedding from Dat Quoc Nguyen's BERTweet pretrained model (which is particularly for sentiment analysis).

3 Data

3.1 Data Description

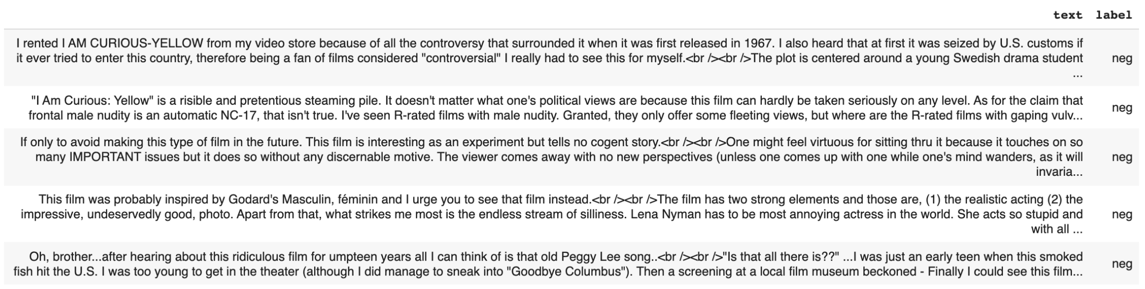
Our dataset comes from Andrew L. Maas's IMDb movie review dataset.(see IMDb Movie Review Dataset Description) provided by the paper Learning word vectors for sentiment analysis [1]. The dataset classifies the movie reviews into positive or negative label according to the given score for the movie. In specific, ≥ 7 out of 10 is labeled as positive and ≤ 4 out of 10 is labeled as negative. Both training and test dataset contains 25,000 examples, in each sub-dataset, positive and negative data are evenly distributed.

Following are some examples about the IMDb movie reviews for positive and negative categories:



text	label
A hit at the time but now better categorised as an Australian cult film. The humour is broad, unsubtle and, in the final scene where a BBC studio fire is extinguished by urinating on it, crude. Contains just about every cliche about the traditional Australian pilgrimage to 'the old country', and every cliche about those rapacious, stuck up, whinging, Tory Brits. Would be acceptable to the Brit...	pos
I love this movie like no other. Another time I will try to explain its virtues to the uninitiated, but for the moment let me quote a few of pieces the remarkable dialogue, which, please remember, is all tongue in cheek. Aussies and Poms will understand, everyone else-well?<br /><br />(title song lyric)"he can sink a beer, he can pick a queer, in his latest double-breasted Bondi gear." <br /><b...	pos
This film and it's sequel Barry Mckenzie holds his own, are the two greatest comedies to ever be produced. A great story a young Aussie bloke travels to england to claim his inheritance and meets up with his mates, who are just as loveable and innocent as he is.<br /><br />It's chock a block full of great, sayings , where else could you find someone who needs a drink so bad that he's as dry as...	pos
'The Adventures Of Barry McKenzie' started life as a satirical comic strip in 'Private Eye', written by Barry Humphries and based on an idea by Peter Cook. McKenzie ('Bazza' to his friends) is a lanky, loud, hat-wearing Australian whose two main interests in life are sex (despite never having had any) and Fosters lager. In 1972, he found his way to the big screen for the first of two outin...	pos
The story centers around Barry McKenzie who must go to England if he wishes to claim his inheritance. Being about the grossest Aussie shearer ever to set foot outside this great Nation of ours there is something of a culture clash and much fun and games ensue. The songs of Barry McKenzie(Barry Crocker) are highlights.	pos

Figure 1: Examples for Positive Categories



text	label
I rented I AM CURIOUS-YELLOW from my video store because of all the controversy that surrounded it when it was first released in 1967. I also heard that at first it was seized by U.S. customs if it ever tried to enter this country, therefore being a fan of films considered "controversial" I really had to see this for myself.<br /><br />The plot is centered around a young Swedish drama student ...	neg
"I Am Curious: Yellow" is a risible and pretentious steaming pile. It doesn't matter what one's political views are because this film can hardly be taken seriously on any level. As for the claim that frontal male nudity is an automatic NC-17, that isn't true. I've seen R-rated films with male nudity. Granted, they only offer some fleeting views, but where are the R-rated films with gaping vulv...	neg
If only to avoid making this type of film in the future. This film is interesting as an experiment but tells no cogent story.<br /><br />One might feel virtuous for sitting thru it because it touches on so many IMPORTANT issues but it does so without any discernable motive. The viewer comes away with no new perspectives (unless one comes up with one while one's mind wanders, as it will invaria...	neg
This film was probably inspired by Godard's Masculin, féminin and I urge you to see that film instead.<br /><br />The film has two strong elements and those are, (1) the realistic acting (2) the impressive, undeservedly good, photo. Apart from that, what strikes me most is the endless stream of silliness. Lena Nyman has to be most annoying actress in the world. She acts so stupid and with all ...	neg
Oh, brother...after hearing about this ridiculous film for umpteen years all I can think of is that old Peggy Lee song..<br /><br />"Is that all there is??" ...I was just an early teen when this smoked fish hit the U.S. I was too young to get in the theater (although I did manage to sneak into "Goodbye Columbus"). Then a screening at a local film museum beckoned - Finally I could see this film...	neg

Figure 2: Examples of Negative Categories

3.1.1 Data Preprocessing

Remove meaningless characters:
The raw dataset contains many useless but confusing pieces for machine learning models. Thus, we processed data-cleaning following the steps below:

- First, we find the raw dataset includes meaningless html symbols, e.g. $< br >< \backslash br >$.

- Second, when comparing to the vocabulary offered by the dataset, we find the words could be word-word, and some phrases like word—word is confusing.

- Third, we also replaced more than on '-' with space.
- Last, to match the vocabulary, we replace upper character with lower character.

Tokenize Documents:
After removing irrelevant pieces for our task, we need to further tokenize document in to word pieces. And we follow the operations below to generate a handleable size of document vectors.

- First, We follow the word token offered by the vocabulary (more than 80,000 word tokens), instead of standard nltk tokenization, because:
  1. it keeps special sentiment tokens, e.g. :), :(, as well as some punctuations, e.g. !, ?;
  2. it offers sentiment value along with the vocabulary and we would like to incorporate it into our doc vectors.

- Second, we remove the English stop words (around 200) to decrease the size of doc vectors.

- Third, we remove the neutral word tokens (with absolute sentiment value less than 1, from -4.5 to 4.5 in total) to further decrease the vocabulary size. And around 20,000 word tokens are kept at last.
  The distribution of sentiment value among word tokens are shown below:

Generate word-count-sentiment document embeddings:
To make use of the knowledge from Andrew L. Maas, et. al, (2011)[1], we make a simplified version of their document embeddings. We first count the number of times each token is appearing in each document (BoW document representation). Then, we multiply the token count of doc vectors (sparse representation) with its corresponding sentiment values offered by the dataset (shape: (1, 18119)).

Generate deep learning document embeddings:
Transformers is one of the most popular method in NLP tasks in recent years. We want to take advantage of this approach by extracting the doc embeddings based on pretrained model. The pretrained model we utilized is BERTweet, which is fine-tuned on RoBerta. [4] The reason for choosing this model is:
1. it keeps sentiment tokens in tokenization phase and
2. it is specifically trained for sentiment analysis using Tweeter dataset.

Here, we explore two ways of generating doc embeddings: [CLS] token embedding and average token embedding.

- **[CLS] embedding:** We take the last hidden state's [CLS] token embedding represents the embedding for the document (shape: (1, 768)).
- **average token embedding:** We take the average, along columns, of all the unmasked token embeddings of last hidden state (shape: (1, 768)).

Embedding summary:

Doc embedding	Size	Train	Dev	Test
word count*emotional value	(1, 18119)	94.96	86.48	85.15
[CLS] token	(1, 768)	85.39	84.4	84.32
Avg embedding	(1, 768)	86.03	85.14	85.34

comparing the three types of embeddings, we identify two weakness of the first word count sparse embedding:

1. the embedding size is too big when storing as a vector.
2. sparse embedding has a large risk of overfitting (the accuracy score drops from 95% to 86%) when the training examples are limited (20,000 training examples and 18119 doc embedding size for each document).

So, considering the above shortages of sparse representation, we decided to drop the sparse representation and keep the other two dense representations with further experiments.

Train, Dev, Test Split:
In this task, we do not need to do any operations on the train, test split. The dataset has already been split into train and test. There are 25,000 samples in training dataset and 25,000 in test dataset. In training dataset, all the positive examples follow all the negative examples. So we shuffle (with seed 0) the training data and then take the first 20,000 example for training and the remaining 5,000 example as development dataset. Then, we apply the above three data preprocessing procedures (word-count-sentiment embedding, [CLS] token embedding, and average token embedding) onto all three datasets.

3.2 Evaluation Metrics

In this task, we choose to use accuracy score as our evaluation metrics:

$$accuracy = \frac{TP + TN}{Total}$$

accuracy calculates all truly classified examples among all examples. We choose accuracy as our scorer because both truly classifying positive and truly classifying negative are of the same importance for us. So, accuracy score is more reasonable compared to F1, which is commonly used in NLP tasks.

4 Models and Experimental Results

The main task of the models is to classify the movie reviews into two groups. Therefore the focus is on classification models. Different classification models from baseline logistic regression to very large BERTweet models were analysed for this project.

4.1 Linear models

4.1.1 Baseline model: Logistic Regression

Since logistic regression is one of the simplest linear models which can be used for classification, it has been used as a baseline model for this task. Logistic regression with cross-entropy loss and L2 penalty with *max_iter* capped to 500 was run on both the [CLS] and average embeddings. This classifies the embeddings into two classes, labels 0 and 1.

Data distribution hypothesis: $h(x) = P(y = 1|x; \theta) = 1/(1 + e^{(-\theta * x)})$
Loss: CrossEntropyLoss+L2: $-ylog(h(x)) - (1 - y)log(1 - h(x)) + ||\theta||_2$

This has worked well as expected, giving a decent accuracy of about 84-85 % for both embeddings.

4.2 Non-Linear models

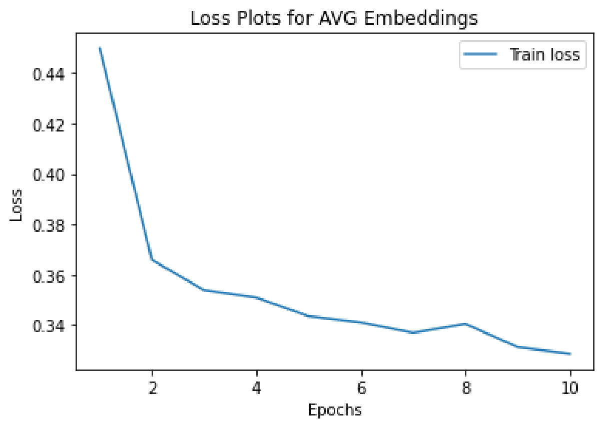
As the classification of 768 long vector into either positive or negative review is a highly non-linear task, we decided to experiment with a few non-linear models like SVM and NNs. Since we're using embeddings for classifications, and not the original text, Naive Bayes is not considered useful here.

4.2.1 SVM

We're using SVM with RBF kernel. SVM is performing poorer than logistic regression here. We're getting 83-84 % accuracy with SVM. This is also computationally very expensive and took the longest training time among the traditional ML models. As our main models of interest are NNs and DNNs, the hyperparameter tuning is kept limited with this model. Also we want to explore the majority based ensemble model, for which we'll need distinct predictions from odd number of models. That is another reason we've used SVM here.

4.2.2 Shallow Neural Network

This is one of the most interesting model that we wanted to try. We started off with simple 3 layer design consisting of linear input layer(768, 128), one hidden layer(128, 32), and output layer(32, 2). We used cross entropy loss with Adam optimizer with learning rate 0.001. We also had to develop custom class for dataset to be used with dataloaders for training and testing. With this architecture, we were getting high training accuracy but poorer test accuracy with only 10 epochs. Thus we identified this as a overfitting problem. To overcome this, we added a dropout layer after the hidden layer. This worked beautifully, boosting the accuracy to 85 %. Adding more linear layers, more dropout layers, and increasing number of epochs didn't result in any further improvement in accuracy. And it took more (25) epochs for [CLS] embeddings as compared to the avg embeedings.



4.3 Ensemble model: Majority Based

This was an interesting experiment to try. We collected the predictions from each of the above three models, and considered the majority of all those predictions as the final prediction. This did improve the accuracy for [CLS] embeddings slightly.

4.4 Deep Learning models

4.4.1 BERTweet-Base (Roberta)

BERTweet-base model is a RoBERTa-base model fine-tuned on 850M English Tweets. [4] It has the same architecture as RoBERTa-base, which contains 12 layers and 125M parameters.

we further fine-tuned this model on our IMDB movie reviews dataset (20,000 samples) for 10 epochs. It reaches a 89.65% accuracy on IMDB test dataset (25,000 samples), which is an obvious improvement from traditional machine learning models (even with ensemble approach).

4.4.2 BERTweet-Large (Roberta)

BERTweet-large model is a RoBERTa-large model fine-tuned on 873M English Tweets. [4] It has the same architecture as RoBERTa-large, which contains 24 layers and 355M parameters.
we further fine-tuned this model on our IMDB movie reviews dataset (20,000 samples) for 10 epochs. It reaches a 95.46% accuracy on IMDB test dataset (25,000 samples), which further improve the accuracy for more than 5% compared to fine-tuned BERTweet-base model.

5 Results and Analysis

Classification Models	Acc(%): [CLS] token	Acc(%): Avg embedding	Num of Params
Logistic Regression	84.42	85.36	768
Support Vector Machine	83.68	84.78	0.02M
Shallow Neural Net	85.13	83.81	0.1M
Ensemble (LR+NN+SVM)	84.42	85.00	0.12M
BERTweet-Base (Roberta)	89.65	-	125M (130 tokens)
BERTweet-Large (Roberta)	95.46	-	355M (514 tokens)

As it is evident from the table, none of the traditional ML models are producing more than 85.5 % accuracy. The classification of these long vectors into two groups is such a non-linear and complex task that the structures of these models are just not enough to capture enough information from the embeddings to train sufficiently. In other words, the traditional models are not able to learn enough of the semantics of the reviews. And training with more data is just resulting in the overfitting. There's a exponential increase in number of parameters from 0.1M to 135M for the deep neural network Bertweet-base(roberta) model for about 5 % increase in the accuracy. And from Bertweet-base to Bertweet-large, there's again enormous increase in parameters from 135M to 355M for another 6 % increase in accuracy. Also the acceptable number of tokens is increased from 130 to 514 tokens to observe this increase in accuracy. This confirms that we need much bigger models for this task if we want the accuracy to be >90 %. While for movie reviews classification, it might not be worth but definitely for other NLP based tasks, this increased accuracy can be crucial. It is wise to leverage the pre-trained models like Bertweet for both developing embeddings and classification.

It is noticeable that for logistic regression, SVM, and shallow NN, the accuracy for average embeddings is better than the [CLS] tokens. This may be because the average embedding takes the average of all the columns rather than just the CLS column and thus has more information to train on.

6 Conclusion

With this project we learned that more complex model structures and larger number of parameters enable the model to fit larger and complex dataset and tasks to reach higher accuracies. Also better document embeddings with proper data pre-processing can help for better learning. Simply training simple models with more data or epochs often results in overfitting and fails as the model architecture can be just not enough to capture complexities of the dataset. All these improvements in embedding, model architecture, parameters are going to demand more compute resources and longer training time. We have to make a trade-off between improved accuracy and increased compute resources.

7 Future Work

Learning from all the experiments, we have identified some more experiments that we can do in future. we could train on 10 %, 30 %, 50 %, 70 %, 100 % training data to investigate the influence of document size on model performance, and we can further see whether we need more training data with transfer learning. For embeddings, we can also explore on how to train an more sentiment based contextualized word embedding/doc embedding to get better results. For ensembling, we can have predictions from more models and also take the weighted predictions. We can also consider working on GPT-3 generative model, which has become popular these days, for classification tasks.

References

[1] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150, 2011.

[2] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1555–1565, 2014.

[3] Usman Naseem, Imran Razzak, Katarzyna Musial, and Muhammad Imran. Transformer based deep intelligent contextual embedding for twitter sentiment analysis. *Future Generation Computer Systems*, 113:58–69, 2020.

[4] Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. Bertweet: A pre-trained language model for english tweets. *arXiv preprint arXiv:2005.10200*, 2020.