Types of Tokens for IMDb Movie Review Sentiment

Jong Heon Han
analyst
jongheon.han
@nyu.edu

Graham Harris
writer
graham.harris
@nyu.edu

Christine Ma
programmer
christine.ma
@nyu.edu

Judy Zhang
researcher
judy.zhang
@nyu.edu

Abstract

In this experiment, five different types of tokens were used to determine what components are necessary for sentiment analysis. Movie reviews from an IMDb data set containing 50,000 reviews were parsed into different tokens, which were then run through a model. By testing all of the tokens, casting to lowercase, removing punctuation, using adjectives only, and lemmatizing, the research team tested what parts of a given movie review are necessary for the sentiment to remain intact. Each tokenization method was run through a machine learning model to evaluate the accuracy of the system on a binary classification model (0 for negative, 1 for positive). The team found that the trial containing all tokens performed the best, likely because each of the subsequent trials removed information important to determine sentiment, such as capitalization and punctuation. The adjectives-only trial performed the worst; since adjectives are not the only part of speech that contains sentiment, it is proposed that more than just one part of speech is necessary for sentiment analysis. All the tokenization methods performed under 90% accuracy, probably due to using a unigram model and data set error. The discovered accuracies lie in the 80-90% range as opposed to state of the art (XLNet as of the writing of this paper) at 96.21%.

1 Introduction

Classification of text is an important feature of natural language processing. Given sequences of text, systems have been built that identify document types, study language patterns, create predictions, and even generate entirely new bodies of text.

However, there are many difficulties when attempting sentiment analysis of a given corpus because human language is complex and interpretable. To simplify the problem, models are typically executed on corpora with strong sentiment, because classification is easier when a corpus undeniably evokes a certain emotion or opinion. Since movie reviews contain bodies of text with strong sentiment, they are a great starting point for analysis. Movie reviews, in this case IMDb reviews, are also usually concise and straightforward, often lacking the length and linguistic complexity of other bodies of text like fiction. Most importantly, movie reviews typically only contain binary sentiment, described as "positive" or "good" reviews and "negative" or "bad" reviews.

2 Problem

Given a dataset of reviews with an assigned binary classified sentiment of positive and negative (represented in the data set as 1 and 0 respectively) our team aims to test different types of tokens on a reputable model in order to analyze the importance of certain types of tokens on sentiment analysis. Several different types of tokens will be compared to discern the best system. Through our analysis, we hope to determine which tokens are necessary to determine sentiment and which are not.

Five different sets of tokens will be used: all

tokens (used as a control), lowercase tokens, lowercase without punctuation, lowercase adjectives only, and lowercase lemmatization. By varying the types of tokens that are fed as input into the model, our team will discover which method leads to the most accurate classifications. After analyzing each tokenization method, we explore what components of a sentence, out of the tokens experimented on, are necessary to accurately describe binary sentiment.

Various tokenizations of a review were then analyzed to explore why one method performed better resulting in the predicted sentiment. Also presented are some ideas for further and continued research.

3 Data Set

The data set consists of 50,000 IMDb movie reviews provided by Stanford University.

ai.stanford.edu/ amaas/data/sentiment/

The set contains a balanced distribution of highly positive and highly negative movie reviews divided into a validation set, test set, and training set. Of the reviews, 25,000 reviews are split into a validation set containing 5,000 reviews and a training set containing 20,000 reviews, randomly distributed by a seed value. The test set contains 25,000 balanced reviews. Also included is a set of all 50,000 unlabeled balanced reviews which were not used.

In the training set, each review has an ID value, a binary sentiment score (1 for positive, 0 for negative) and the movie review text. The test set is the same format as the training set and is used to evaluate the system.

4 Model and Training Details

The structure for implementing and training the model is from the SciKit Learn Python library and is written in Jupyter Notebook. Reviews are first passed through a count vectorizer in order to translate the tokens into a machine-readable format. The count vectorizer reads through all the reviews in the data set and assigns a frequency value based on the number of instances of the token in the corpus.

After, the returned vector is run through a TF-IDF algorithm. For a word t in a document d of a set of D documents the TF-IDF is calculated as such:

$$tf\text{-}idf(t, d, D) = tf(t, d) * idf(t, D)$$

Where tf and idf are defined as:

$$tf(t, d) = log(1 + freq(t, d))$$

$$idf(t, D) = log(\frac{N}{count(d \in D : t \in d)})$$

The purpose of performing TF-IDF on the vectorized tokens is to compute the importance of a word in a given movie review. TF-IDF compares the number of instances of a word relative to how important the word is in each movie review thus assigning a weight for each token in the review.We chose to use TF-IDF because our focus on tokens as a unigram allows us to determine the relationship between a singular word and sentiment analysis.

The movie review training set was divided into 80% training data and 20% development using the same seed each time to provide consistency between each model training. The weights are then transformed into an array that is used to train the model using logistic regression.

Logistic regression is a method of classifying data using a binary system. The model applies the associated TF-IDF weighted tokens and then predicts the dependent variable, in this case negative or positive through a sigmoid/logistic function to the data set. The prediction is a binary classification of either 0 indicating a negative review or 1 indicating a positive review. The threshold for a positive review is 0.5; any number lower than 0.5 is considered negative.

Afterwards, the accuracy and loss function of each model was plotted to determine the number of total iterations needed. We looked at where the lowest point the training and development/validation set are to determine the necessary number of epochs. Then for consistency among each experiment, we chose to train each model at twenty epochs. All of these steps were performed for each tokenization model.

5 Experiment

5.1 Tokenizers

Five different token types were used on both the training and test dataset. By building on a tokenization and refining each method, the hypothesis is that the vectorizer will be able to more accurately count the number of instances of a word, which will impact the TF-IDF score.

The base case is all tokens unchanged which have been split by the NLTK tokenizer. All tokens include alphabetic characters which include lowercase and capitalized characters, and punctuation. By using all of the tokens as the baseline, it becomes clear how the alterations of each token impacts discovered sentiment.

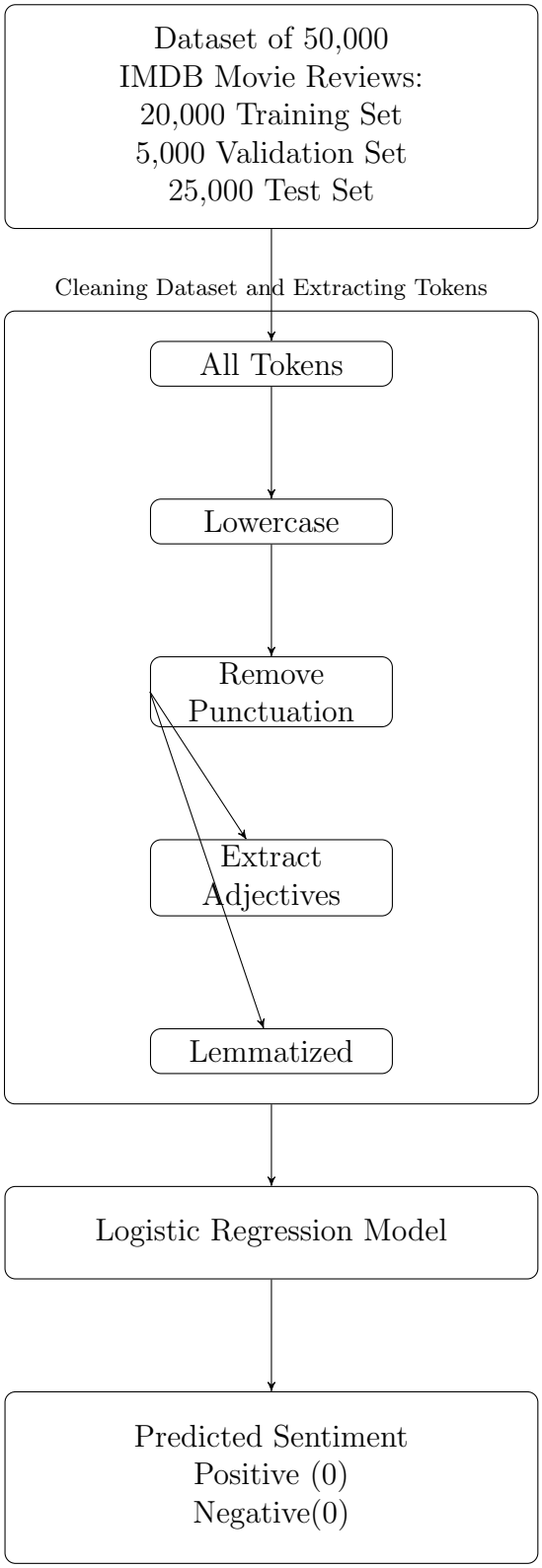
The second tokenization method is using lowercase tokens. From the baseline tokens, the alphabetic characters were cast to lowercase using Python's native String function. In this occasion, lowercase tokens also include punctuation. By our hypothesis, in the base case "the" and "The" may be considered separate tokens, but not in the lowercase method.

The no punctuation method is an extension of the lowercase method where using NLTK's regex tokenizer, all forms of punctuation were removed. Since punctuation was removed, it also resulted in words such as "it's" being split into two tokens "it" and "s". Removing punctuation would decrease the number of non-word tokens, which may increase the system's ability to understand sentiment.

The adjective uses the tokens outputted by removing punctuation and applies the NLTK POS tagger to determine which words are adjectives. Each review consists of the qualities, lowercase and punctuation removal.Then all tokens that are not labeled as adjectives are removed. The tokens consist of adjectives in a superlative, comparative, or base form. This could be done with any selected part of speech, adjectives were only chosen because they are descriptors and may hold relevant sentiment information. A further research question may analyze the accuracies of each part of speech or combinations of them for sentiment analysis.

The last tokenization method is lemmatization. Lemmatization is the process of deriving a root for each token - the word "companies" is lemmatized to "company", and the word "sang" has "sing" as its root. The lemmatization is applied using the tokens produced by removing punctuation. So the tokens include words that have been cast lower case, no punctuation, and lemmatized. Similarly, stemming is another method of reducing the number of different types of tokens present in the system which we considered extending research on. By reducing the variation of tokens, it may lead to increased performance.

5.2 Pipeline



5.3 Data

The first step of the experiment was to clean the dataset of miscellaneous symbols, characters, and number values. Next, the reviews were tokenized by the specifications previously defined. The tokens for each trial were then written to a CSV file that contained one row for each tokenized review across five columns, one column for each method of tokenization.

Then, the model was trained and the loss per iteration was determined. After analyzing the number of epochs necessary, the number of universal iterations chosen was twenty. After training the model with the development set, the test data was passed through the pipeline to predict the binary sentiment. Furthermore, the probability of each incorrect positive and incorrect negative prediction was recorded for use in error analysis. The binary classification predicted by the model was then compared to the answer key to discover the accuracy. The discovered accuracies for each token type are presented in Table 1.

Tokens	All Tokens	Lowercase	No Punctuation
Accuracy	88.18 %	87.82 %	87.90 %
Number of Incorrect Predictions (out of 25,000)	2956	3046	3026

Tokens	Adjectives	Lemmatized
Accuracy	82.80%	87.88 %
Number of Incorrect Predictions (out of 25,000)	4300	3030

Table 1: Test Dataset Performance.

After the data was collected and analyzed, the reviews that were inaccurate by a margin of 90% were compiled for analysis. The top fifty words with the highest frequency were also extracted. Additionally, reviews where only one type of tokenization method predicted incorrectly were analyzed as well.

6 Results and Analysis

The accuracy of each trial is under 90%, which is around the range of acceptable machine error for the system. It is worth mentioning that 5.81% of the reviews (1,453 reviews) were misclassified by all five tokenization methods. Therefore, the

variance of accuracy for each trial is around 7-13%. Since the trial with all of the tokens is the most trivial system, it is a good point to compare the other four trials against. By comparing the base case of the all tokens method against the failures of the other systems, it becomes more clear where the sources of error were.

6.1 Lowercase vs. All Tokens (Base Case)

The "all tokens" trial split all of the reviews by space, but did not perform any operations on the tokens. Opposite to our hypothesis, this trial performed the best of all five trials. It was discovered that when this trial correctly predicted positive sentiment, every other tokenizer wrongly predicted negative sentiment. In these cases where this was the only tokenization method that predicted correctly, our team focused on the effect of capitalization, since the next trial removed this aspect. Of all the tokens in the data set, about 10% of them were capitalized - these tokens were all cast to lowercase in all subsequent trials.

The most frequent capitalized tokens within positive classification in the trial were "The" (414 instances) and "This" (138 instances). Therefore, our theory is that capitalization of these two words had a positive connotation which led to a prediction over the .50 threshold. In comparison, the lowercase versions of these tokens predicted a value in the .45-.49 range.

The lowercase trial performed slightly worse than the first trial, by a margin of 0.36%. This may be because words that are capitalized, especially those where every word is capitalized, may be weighted more. For example, the review with ID "8570_8" contains a fully uppercase "HAS" and "NOT" to indicate emphasis on certain phrases. By casting to lowercase, these tokens are instead interpreted as lowercase tokens, which may not carry as much sentiment weight due to the commonality of their lowercase forms.

6.2 No Punctuation vs. All Tokens

The trial without punctuation also performed slightly worse than the all tokens trial, a margin of 0.36%, but better than the second trial (lowercase) by 0.06%. Of all the reviews, there were only 4 that this trial uniquely predicted wrong. Of those 4 reviews, this trial predicted within a range of 0.45-0.49, whereas both the all tokens and lower case predicted within a 0.50-0.53 range. Since around 10% of the tokens in each review were a period, our team concluded that periods are a positive token with about 0.01 positivity. So when removed, the model predicted more negatively.

6.3 Adjectives vs. All Tokens

The adjectives-only method performed the worst of all the tokenizers at 82.80%, which is 5.38% worse than the baseline. This may be because just one part of speech is not enough for the system to accurately be able to determine sentiment. For example, there is a very large difference between "good" and "not good," but since tokens like "not" were absent (as well as every other non-adjective token), important sentiment information was missing.

Since so much context is missing, it is not surprising that the adjective trial would not be able to determine nuance in the movie reviews. It had the largest number of reviews that were incorrect by a margin of error of 90%, at 189 reviews. To put that into perspective, the all tokens trial had 58, the lowercase trial had 87, the no punctuation trial had 88, and the lemmatization trial had 52. The adjective trial had 114% more drastically wrong predictions than the no punctuation trial, which is a very distant second.

For an anecdotal example, one review is provided from the sample of adjective-only reviews that were wrong by an error of over 90%. The original review (ID "7139_10") is

> "have seen Chasing the Dragon several times and have enjoyed it each time . The acting was superb . This movie really makes you realize how one bad choice at a weak moment can change your life ."

The tokens that the system performed sentiment analysis on were "several," "bad," and "weak". These adjectives were used to describe the contents of the movie, not how the reviewer felt about it, but the system could not distinguish between those two contexts.

It is also important to note that the tokenizer did not label the word "superb" as an adjective in the example above. This is crucial to note for two reasons. Firstly, the word "superb" indicates positive sentiment and is just one example of where tokenizers influence the ability of the model to determine sentiment. Secondly, by removing all tokens except adjectives, only 8% of the tokens were actually analyzed. In this example, the word "superb" is only 2% of the entire review, but if it was included in the recognized adjectives, it would account for 25% of the tokens analyzed. Since removing all parts of speech except adjectives reduces the sample size so drastically, just a small amount of error can lead to an entirely different predicted sentiment.

6.4 Lemmatization vs. All Tokens

The lemmatization trial performed similarly to the lowercase and no punctuation trials - it did 0.06% better than the lowercase trial and 0.02% worse than the no punctuation trial. Since it performed similarly to those, our team concluded that the same problems that plagued those trials also affected the outcome of this trial.

Where the lemmatization method shined, however, is that it had the lowest number of reviews that were off by a margin of over 90% error. There were only 52 reviews in this category, which compared to the second best trail (the all tokens trial) at 58, is 11.5% more successful. Since both the lowercase and no punctuation trials did much worse in this area, with 87 and 88 reviews incorrect by a 90% margin respectively, our team concluded that lemmatization reduces extreme variation in the system. By reducing words to their roots, the system is much less likely to overcount the sentiment of a particular token.

6.5 Sources of Error

Some possible sources of error are now proposed. As with many language systems, some of this analysis may be subjective due to different interpretations of the meanings of words.

One source of error is that only one experiment was done with each tokenization method. The first three trials were all more objective, since there are very few ways to split tokens by space, cast tokens to lowercase, and remove punctuation. However, the lemmatization and adjective only methods are much more subjective than the other three methods. For all tokenizations in the experiment, our team used the NLTK Python library. The lemmatization method used the NLTK lemmatize and the adjective method used the NLTK part of speech tagger. Since the team restricted ourselves to just one NLP library, there are errors associated with that library. Consider the example from the adjective experiment analysis - the word "superb" was not properly tagged as an adjective, so the system missed a crucial token. By diversifying the libraries used, this kind of error may have been avoided or reduced.

It is also worth noting that the reviews are not all consistent in structure. IMDb reviews are submitted by users who use slang and improper grammar, as well as inappropriate and foreign words. For example, the review with ID "1510_8" is about a foreign film called "Un Gatto nel Cervello" (titled "Nightmare Concert" in English but literally translated to "A Cat in the Brain" from Italian) and the review contains the names of the director and actors, as well as several grammatical errors. All of these factors introduce noise into the model that reduces the capacity of the system to determine sentiment.

There is also inconsistency in the labeling of the reviews in the answer key. Even though the reviews are subjective, it seems like the data set may not have been annotated 100% accurately. For example, the review with ID "426_9" reads,

> "This is definitely one of the best Kung fu movies in the history of Cinema . The screenplay is really well done is not often the case for this type of movies and you can see that Chuck one of his first role a great actor . The final fight with the sherif deputy in the bullring is a masterpiece !"

To the research team, this seems to be an overwhelmingly positive review, specifically "one of the best" and "masterpiece". Instead, the answer key labeled this review as negative.

All of these sources of error combined led to the inaccuracy of the system. The team intentionally chose a simple logistic regression

model and a very basic set of tokenizations; the idea was to look for the simplest methods and what results those methods would achieve. Even though the data set and program are relatively small, our results provide interesting pointers as to why the BERT models work better than this one and what obstacles exist when processing natural language. The most fundamental machine learning models already achieve results significantly better than ours, which is impressive when considering the future trajectory of the field.

7 Conclusion

The outcome of this experiment was surprising. The research team expected that the lowercase, no punctuation, and lemmatization trials would do better than the all tokens trial. Since those three trials reduce the amount of "noise" in the system, such as capitalization, punctuation, and different forms of words, our hypothesis was that the model would be able to more accurately determine sentiment. However, through our analysis, it became clear that some of the tokens removed, such as the periods at the end of sentences, were essential.

7.1 Further Research

There are many opportunities for further research. Several further research topics are proposed based on observations made during this experiment.

One idea is to try different tokenizers of all one type of tokenization to try to figure out the best tokenizer of that method. While part of speech taggers may not have a 100% accuracy rate, an experiment could be done by using more than just the NLTK POS tagger.

Another is to use a model with higher order than just unigram. Our model does not take negation into account, such as "not good" or "terribly good". By using a bigram or higher dimensional model, the system may achieve higher accuracy since it can make contextual calculations.

The system could also be replicated on reviews from different source material, such as Yelp or Amazon. It could also be worthwhile to find a data set that is "cleaner" or more refined than the IMDb set. The mislabeling of reviews introduced variance between the model, which may have skewed accuracy.

References

Bonfil, Ben, and Ieva Staliunaite. 2017. Breaking Sentiment Analysis of Movie Reviews. *Proceedings of the First Workshop on Building Linguistically Generalizable NLP Systems*, pp. 61-64. https://www.aclweb.org/anthology/W17-5410.pdf.

Johnson, Rie, and Tong Zhang. 2016. Supervised and Semi-Supervised Text Categorization using LSTM for Region Embeddings. https://arxiv.org/pdf/1602.02373.pdf.

Raford, Alec, Rafal Josefowicz, and Ilya Sutskever. 2017. Learning to Generate Reviews and Discovering Sentiment. https://arxiv.org/pdf/1704.01444.pdf.

Sun, Chi, et al. How to Fine-Tune BERT for Text Classification? https://arxiv.org/pdf/1905.05583.pdf.

Yang, Yi, and Jacob Eisenstein. 2017. Overcoming Language Variation in Sentiment Analysis with Social Attention. *Transactions of the Association for Computational Linguistics, vol. 5*, pp. 295–307. https://www.aclweb.org/anthology/Q17-1021.pdf.

Yang, Zhilin, et al. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *NeurIPS 2019, no. 33rd Conference on Neural Information Processing Systems.* https://arxiv.org/pdf/1906.08237.pdf.