Sentiment Analysis on Amazon Fine Food Review Dataset

Xinyi Li, Hongzhi Liu, Sheng Wang, Siyu Zhang[1]

[1]University of Wisconsin-Madison

rates are skewed towards 5 stars: merge 1~3, 4～5

Abstract

In this project, we conducted a sentiment analysis for fine food reviews using natural language processing. A recurrent neural network was built to predict the user rating for fine food according to the comment. Model hyperparameters were selected according the cross validation analysis. Overfitting issue was discussed for this model. A precison/recall curve was plotted based on the model output.

1 Introduction

Sentiment analysis is an automated process of understanding opinions from natural language. Nowadays countless reviews are available on internet, but almost all of them are unstructured data, making direct analysis difficult and time-consuming. Sentiment analysis can help address this issue by quantifying the information in unstructured data, thus the polarity hidden behind the texts can be extracted for further analysis.

Recently, sentiment analysis has gained much popularity and has been used to resolve many practical issues, such as marketing analysis, customer service, etc. In this report, we apply sentiment analysis on Amazon fine food review dataset from Stanford Network Analysis Project (SNAP) which contains $568,454$ reviews from more than $200,000$ users to study customer behavior and predict the ratings. There are many methods to implement sentiment analysis when our goal is to predict the level of polarity of opinions (thus can be viewed as a classification problem), such as Logistic Regression, Support Vector Machine, Neural Network. Since the model needs to process language input, we chose Recurrent Neural Network(RNN) with embedding for this problem set.

2 Feature Selection

The data we use is Amazon fine food review dataset from SNAP. It contains $568,454$ reviews from more than $200,000$ users which are collected between Oct 1999 and Oct 2012. Each data instance consists of 10 variables including product ID, user information, profile name of the user, number of users who found the review helpful(HelpfulnessNumerator), number of users who indicated whether they found the review helpful or not(HelpfulnessDenominator), rating, timestamp for the review, brief summary of the review, and plain text review. The statistics of major variables are shown in Table 1.

Num. of Products	74258
Num. of Users	256059
HelpfulnessNumerator Average	1.743817
HelpfulnessDenominator Average	2.22881
Score Average	4.183199
Time	Oct 1999 - Oct 2012

Table 1: Summary for Variables in Amazon Fine Food Review

Among all the features provided in the data set, we select HelpfulnessNumerator, HelpfulnessDenominator, summary of reviews and text review to predict the ratings of a given product by this user. Other variables are considered irrelevant here.

consider time

But summary and reviews are not ready to be fed to the model directly, to structurize text information, we use a skip-gram model to do word-embeddings. The strength of this approach is that it projects texts to a lower-dimensional vector space while capturing the semantic similarity among words. Considering that the longest text reviews contains 3507 words, but the majority of the text review length are below 300 (see Fig.1), we can pad the review sequences at 300 to largely relieve computation challenge while still maintain accuracy. Similarly, as shown in Fig.2, most of the review summaries has length under 50, so we choose to pad the summary sequence at 50. Another detail in feature extraction is that there are tens of thousands of words with extreme low frequency, as shown in Fig.3 & Fig.4. Including these low-frequency words would offer us little benefit in text vectorization but lead in much computational burden. Hence, we only keep the $30,000$ most frequent words to effeciently extract text feature.

By looking at the distribution of ratings shown in Fig.5, we can see that it's highly skewed towards 5-star rating, which takes up to 63.9% of all the ratings. Due to the un-

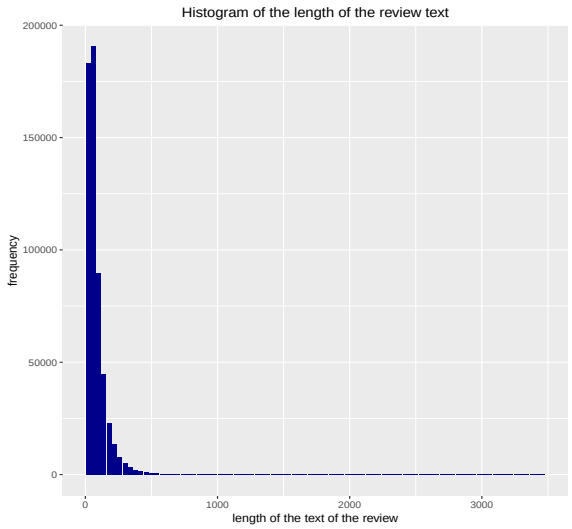

Figure 1: Distribution of Length of the Reviews. X axis is the length of Review for each user, and y axis represents the frequency for each Review length.

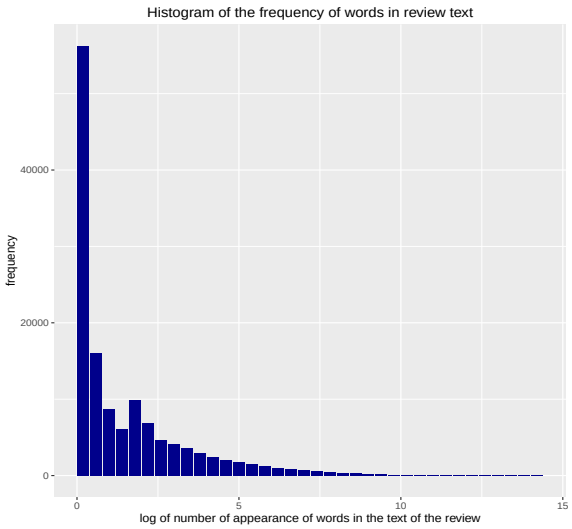

Figure 3: Distribution of Word Frequency of Text in Review. We count the total number of appearance of each word in Review, and then take logarithm transformation to better view the distribution, and y axis denotes the frequency for each log of total appearances.

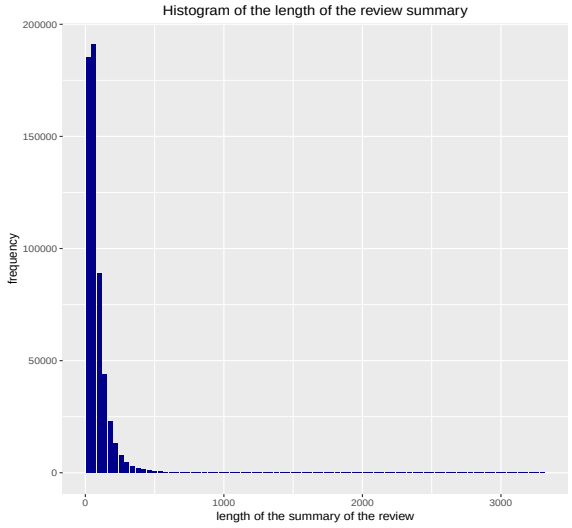

Figure 2: Distribution of Length of Summary. X axis is the length of variable Summary for each user, and y axis represents the frequency for each Summary length.

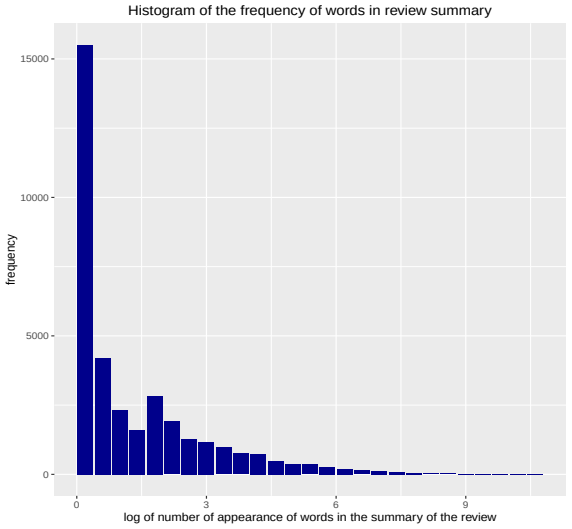

Figure 4: Distribution of Word Frequency of Summary. We count the total number of appearance of each word in Summary, and then take logarithm transformation to better view the distribution, and y axis is the frequency for each log of total appearances.

balanced distribution of the rating, combined with the fact that the scores given by users can be quite subjective and uncertain, we merge ratings 1-3 as ”bad” reviews, and ratings 4-5 as ”good” reviews. In this way, the problem setting can better mimic the false feelings common in many people that the difference between 3-star and 4-star (from bad/neutral to good) is much bigger than that between 4 and 5(among good reviews) or between 2 and 3(among bad reviews). Now our prediction problem becomes a binary classification problem.

3 Model Selection and Empirical Evaluation

We built a model based on user review, brief summary of the review and the information whether the review is useful. The scheme of the model is shown in Fig.6. Review and summary contains text information which needs to be processed. We first converted such text information to lower case and selected 30000 most common words for each as the library for one-hot encoding. Then all reviews and summaries were

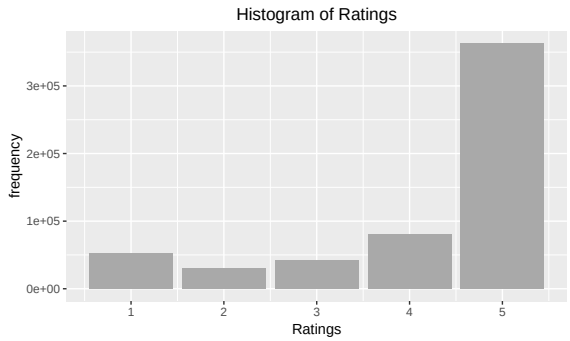

Figure 5: Distribution of Ratings

translated to sequences of one-hot vector according to preset library. Words which do not exit in the library were convert to vectors of 0. Before inputted into the model, sequences were padded to the same length. Review sequences were padded at length 300 and Summary sequences were padded at length 50.

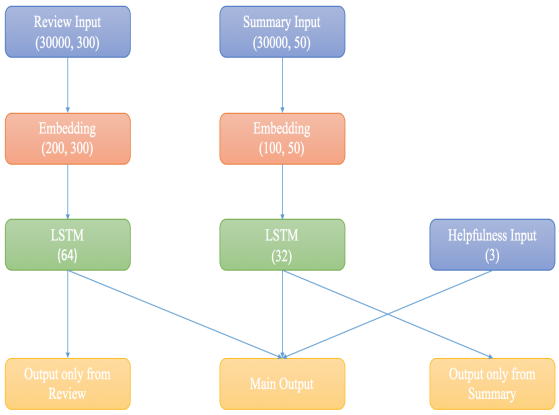

Figure 6: Structure Diagram of the Neural Net.

The preprocessed review sequences were inputted into an embedding layer and trained into sequences of vectors with 200 dimensions. The embedded sequences were used as the input of the LSTM layer with 64 hidden units. 20% units were dropped out for inputs and recurrent state. The output of this LSTM layer were trained to a binary output using sigmoid function as the activation function and binary crossentropy as the loss function.

The same training model was applied on summary data. Since summary data contains less information. Only 32 hidden units of LSTM layer were employed here. A binary output was produced based only on summary data. In addition, the outputs of both LSTM layers were combined along with "helpfulness" data as the input of a third perceptron. The "helpfulness" data are processed from number of users who found the review helpful(HelpfulnessNumerator) and number of users who indicated whether they found the review helpful or not(HelpfulnessDenominator). In addition to the above two features, the ratio between them are calculated as the third feature for this "helpfulness" data. The output of this third perceptron was used as the main output of this model.

In summary, three outputs were produced from this model based on review, summary of the review and the "helpfulness" information. One of the outputs is the prediction only based on the review data. Another is only based on the summary data. The main output of this model learns information from all input data.

Most hyperparameters of the model are selected from detailed feature analysis and previous experience of natural language processing. The model was fitted on the training data using stochastic learning. To better utilize the parallel processing power of GPUs, we selected batch size as 500. The cost of training each instance using GPU with respect to the batch size is plotted in Fig.7. Considering the total size of the dataset(568,454), 500 is also an appropriate selection for batch size. Adam was selected as optimizer with learning rate 0.001.

Figure 7: Batch Size vs Cost for Training Each Instance.

To avoid underfitting, we also added another LSTM layer for each recurrent network in this model and checked the corresponding performance. Comparison of the accuracy of 1-layer-LSTM model and 2-layer-LSTM model are listed in Fig.8.

For this comparison, we randomly selected 100000 data to run cross validation. Number of epochs were chose as 2. From Fig.8, we can observe that adding a LSTM layer does not significantly increase the fitting accuracy. For simplicity, we only employ 1 LSTM layer in this model.

We also calculated accuracy curve for main output with respect to number of epochs as plotted in Fig.9. In this simulation, we utilize all the data from the dataset. 80% data were chosen as traing data and the remianing were used as validation data.

In this plot, accuracy of validation set reaches a plateau when number of epochs is 5. However, the accuracy of the training set still increases. It indicates that when the number of epochs is larger than 5, model overfits the data. Although

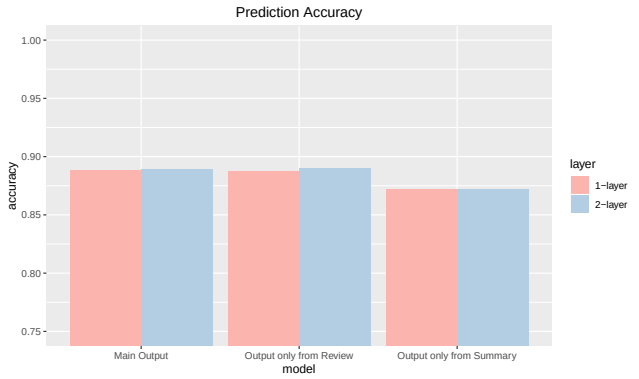

Figure 8: Prediction Accuracy of Two Candidate Models. We compare the prediction accuracy for 1-layer-LSTM model and 2-layer-LSTM model, and we measure the differences based on three kinds of output: main output, output only from Review, and output only from Summary.

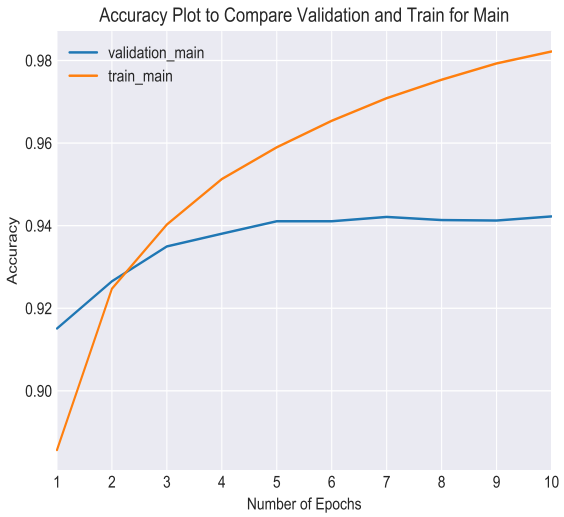

Figure 9: Validation and Train Accuracy for Main. The orange line represents train accuracy, and the blue line denotes the validation accuracy.

they are not plotted here, accuracy of outputs from only review data and only summary data have the same behavior.

4 Data Analysis

In this section, we further analyze the performance of the three in previous sections: one based only on Text Reviews, another based only on Summary, and a third output uses all the information in Text Reviews, Summary and the information whether the review is useful.

We first look at the accuracy of the three models on the test set shown in Fig.10. When the number of epochs is fixed at 5, the accuracy of the main output outperforms the other two. This comparison indicates that there exists useful information hidden in the HelpfulnessNumerator and HelpfulnessDenominator.

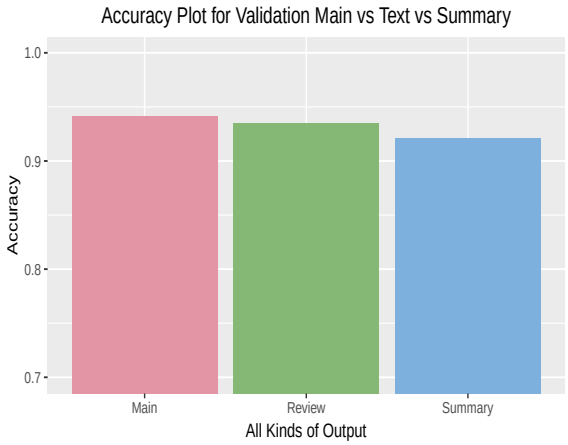

Figure 10: Training Accuracy Plot for Main vs Review vs Summary. We Compare the training accuracy using three different inputs: Main, Review, and Summary. The blue line represents Main output, the orange line denotes Review output, and green line corresponds to Summary output.

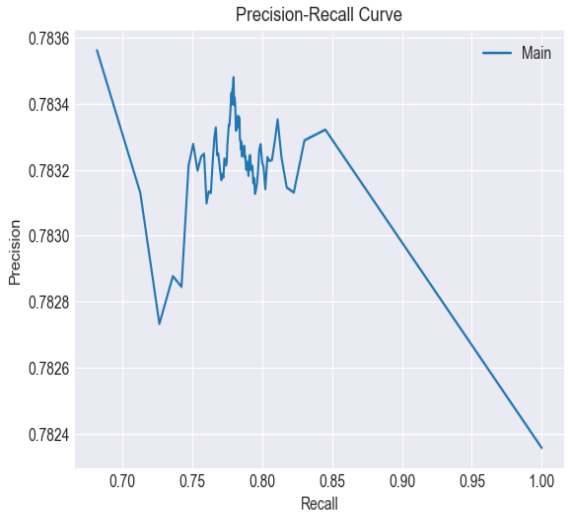

Figure 11: Precision/Recall Curve. This curve is based on the Main output.

Because of the unbalanced nature of this dataset, the accuracy measure can sometimes be misleading. Here we also include a Precision-Recall Curve, which can not only give us the full picture of the performance of the model as the threshold changes, but also are robust to the skewness of classes. From Fig.11 we can see that the area under this PR curve is close to 1, indicating that our model is well-trained and offer much predictability.

The accuracy of the main output reaches 94%. This high accuracy is still informative in spite of the high skewness of the data, because the ratio of the "good" reviews accounts for 78%, which is much lower than our 94% accuracy (which means that the a "naive" classifier that classifies all data

points as positive can only reach a 78% accuracy).

It is also worth mentioning that our method is efficient when working with this huge dataset($568,454$ reviews) containing sequence data and dealing with the natural language processing problem. The entire process to train all the five epochs takes only 32 minutes using current available GPUs.

5 Conclusion

In this project, we developed an efficient RNN network to conduct sentiment analysis based on user reviews. In the future, we hope to refine the model to be able to predict the $5\sim$scale user ratings. This may require a more complicated model and more careful parameter selection. We are also looking forward to adding more meaningful features like the length of review to our model to enhance the performance.