Predicting Usefulness of Yelp Reviews

Annie Yang, Weiqi Pan, Fuyu Zou, Bing Li

January 23, 2019

Abstract

The Yelp Dataset Challenge datasets are publicly available for machine learning projects. They wish to find interesting trends and patterns in the data they have accumulated. In this study, we are interested in predicting the usefulness of Yelp reviews based on review texts and user/business history, such that Yelp can provide recommendation from newly posted useful reviews and improve user experience. To classify usefulness of reviews, we implemented Bag-of-Words (both bigrams and unigrams) and GloVe to generate text representation features and built neural networks including Vanilla Neural Network, Convolutional Neural Network (CNN) and Bidirectional Recurrent Neural Network (Bidir-RNN). To improve the performance of prediction, we also added meta features that are extracted from review and user dataset. By comparing the results among the models we find that the Bigram model and bidirectional LSTM provide the best performance in this binary classification task.

1 Introduction

The data used in this project was downloaded from Yelp Dataset Challenge Round 12 [1]. There are two major datasets included in this challenge, which are the text dataset including the reviews, businesses, and other text information; and the image dataset that includes the pictures users uploaded online. Our group focuses on understanding text dataset and predicting the usefulness of reviews, specifically for restaurant and food business.

Online review plays an important role in information dissemination and could further influence user's decision. However, a user may only read a limited number of reviews before making a decision. An important aspect to the success of a social network which lets its users search and write reviews is to identify which reviews should be promoted to users. From this point of view, Yelp introduced voting option for its reviews. Users could vote "Useful", "Funny" or "Cool" for reviews. These votes could reflect user's thoughts and indicate the quality of reviews.

However, for new reviews or businesses with low traffic, this option is not so applicable. Thus, predicting the usefulness of reviews is a question worth thinking. If usefulness of a review is predicted in advance, fresh reviews with high quality can be recommended to users sooner, and users could have useful information from most updated reviews. For businesses, they can also gain insights into their products and services timely.

2 Related Work

There have been several projects on Yelp reviews dataset [2] [3] [4]. However, most of them focus on using traditional classification methods such as SVM [2], Random Forest [2], Logistic Regression [2] and Topic Models [3]. And some of existing research only use text features in prediction [4]. Our project provides a deep learning perspective by applying some deep learning methods in text processing, which are Bag-of-Bigrams and Vanilla Neural Network, Convolutional Neural Network(CNN), and Bidirectional LSTM, with meta features added in to improve the performance of prediction.

Bag-of-Bigrams is a method based on the n-grams theory that it creates group of vocabulary with 2 words based on the frequency. A Bag-of-Bigrams representation is much more powerful than bag-of-words, and in many cases proves very hard to beat [5]. Because it could change the scope of the vocabulary and allow the capture more information from the text.

CNN on text classification was proved to have a promising result on the text sentence classification recently. Even a simple model with static vectors (CNN-static) performs remarkably well, giving competitive results against the other more sophisticated models. The CNN-non-static model, which allows pretrained vectors to be fine-tuned for each task, could further improve the model performance [6, 7].

Recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed graph along a sequence. This allows it to exhibit dynamic temporal behavior for a time sequence. Unlike feed forward neural networks, RNNs can use their internal state (memory) to process sequences of inputs [8]. The two most successful RNN architectures for sequence learning stem from two papers published in 1997, which introduced the Long Short-Term Memory method and the bidirectional RNN model. These two innovations have been successfully combined into Bidirectional LSTM for further applications [9, 10, 11].

3 Dataset

3.1 Overall Description

All the data we used are publicly available online in JSON format [1]. We have access to information about the reviews, businesses being reviewed and users posting the reviews.

In the yelp review dataset, there is a total of 5,996,996 reviews, with variables of review id, users id, business id, rating stars, date, review text, and votes of funny, cool, useful. Besides the texts, we also add text length as a meta feature into the input of the algorithms, since the text lengths seem to be highly correlated with the outcome as in Figure 1.

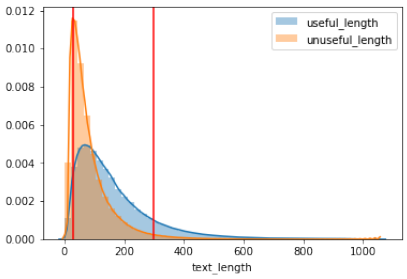

Figure 1: Text Distribution Plot by Useful Label

In the business dataset, there is information about the actual business being reviewed (total 42153 businesses), with features such as the average aggregate star rating, the business category and the total review count. Considering the different properties in different business categories, we only included businesses that are categorized as food or restaurants in our analysis. The business ID for restaurants and food was extracted from the business dataset and merged to the review dataset above in order to only include the review samples for restaurants/food business.

We also made use of the user dataset that includes 252,898 users who posted the reviews. User dataset consists of 21 user features, such as the user's friend mapping, review counts, fans and so on, which could be significantly related to the usefulness and thus improve the prediction performance.

3.2 Data Pre-processing

First, we categorized the outcome of usefulness votes into binary groups. The original usefulness votes range from 0 to 1,234 with a distribution heavily left-skewed that about 75% of the reviews have votes 0 in average. To preserve most information, we decided to set the cut-off points at 1 and 9 and we will compare results for both cutoffs later.

Next, we filtered samples by selecting reviews from 2007 to 2018 (only few reviews were posted from 2004 to 2007), and text length between 30 and 300 (inclusive). Information provided by short reviews is limited and long reviews may introduce some noise words and information. Additionally, when looking at the text length distribution plot (Figure 1) by usefulness label, we found that most of the reviews fell into this range.

We then did some processing before feeding the review texts into the algorithms. We lower cased the text, removed stop words, removed punctuation except question marks and exclamation points from the word corpus. Then we converted word to sequence with padding to the same length of 300.

3.3 Feature Extraction

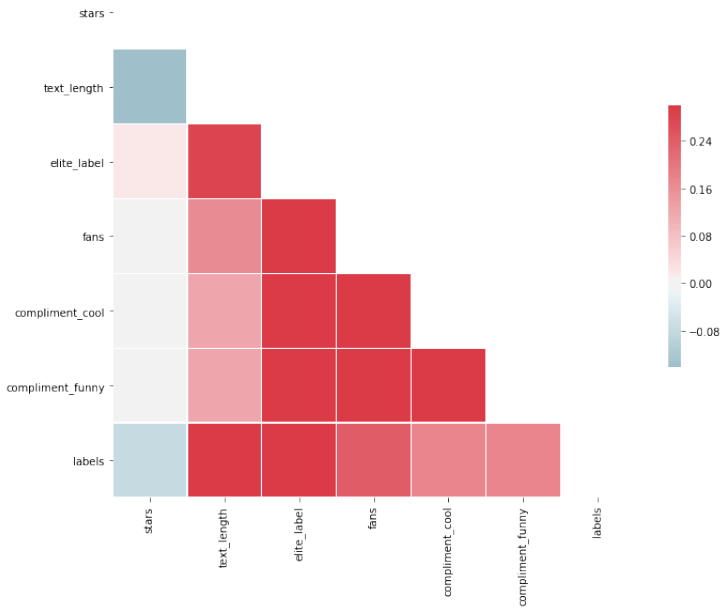

Figure 2: Correlation Matrix of Review and User Dataset

To further improve our model, we explored features that could contribute to the usefulness votes. Figure 2 shows the correlation matrix between useful label (labels represented in the Correlation Matrix) and some potential meta features. The six meta features are highly correlated with usefulness votes, which are star and text length from the review dataset; elite label, fans, compliment cool and compliment funny from the user dataset. So we included all of the six variables into our model as meta features.

To become an Elite Yelper, users need to write as many useful and unbiased reviews as possible. So it is reasonable to take advantage of the historical data of users such as elite and fans information, as well as the historical data of compliment fun and cool which are the counts of fun and cool compliments received by the user.

4 Methods

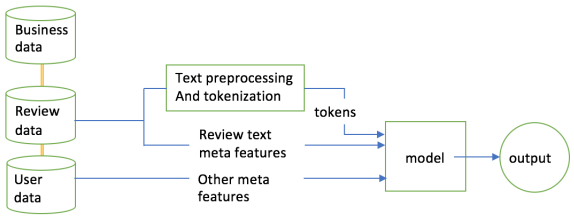

Figure 3: Pipeline

4.1 Bag-of-Words - Unigram and Bigram

Instead of using a simple unigram model, we included both uigrams and bigrams. We converted reviews into a numeric matrix of token counts, where each row is a review and each column is a token (unigram or bigram). In this scheme, each token is treated as a feature. Compared to unigram model, including bigrams of words allows the model to extract more information from reviews, and some information about the structure of words.

With CountVectorizer technique, we extracted top 1,200 words as text features (frequency), and fed them into the vanilla neural network either with or without meta features. The model structure is summarized in Table 1.

Table 1: Model structure for vanilla neural network with Bigrams

Structure	Value
Hidden layers	2
Hidden layer size	128, 64
dropout	only text features: 0.5 ; text + meta features: no dropout
Batch Normalization	
Activation Function	ReLu
Optimizer	Adam: Learning Rate = 0.001

4.2 CNN and Bidirectional LSTM

4.2.1 GloVe embedding

GloVe is another method for learning word representations.[12] The objective of GloVe algorithm is to measure the co-occurrence statistics from a corpus.

Initializing word vectors with those obtained from an unsupervised neural language model is a popular method to improve performance in the absence of a large supervised training set. We utilized the publicly available GloVe weight matrix as the embedding matrix in this project. Specifically, we used *glove.6B.100d.txt*, which was trained on Wikipedia and Gigaword 5. Words that are not included in the set of pre-trained matrix are initialized randomly [12]. By implementing GloVe in our model, we were able to leverages statistical information by training on the nonzero elements in a word-word co-occurrence matrix, rather than on the entire sparse matrix or on individual context windows in a large corpus.

4.2.2 Convolutional Neural Network(CNN)

In this project, we also applied CNN in text sentence classification. Specifically, we applied the CNN-non-static model that could further improve CNN [6]. We used pre-trained GloVe co-occurrence matrix to initialize the weight of embedding.

There are 3 convolutional layers and max-pooling layers in the final training model. The regularization is set with dropout rate at 0.3 in each convolutional hidden layer to prevent over-fitting. After concatenate the meta input and the output from convolution, we added a dense layer to output the final probability. Each convolution layer was in size of 128, with a filter size of 5, stride size of 1 and "valid" padding method. Each layer used activation function ReLu except for the last layer that used sigmoid activation function. The model architecture is attached to the appendix.

One advantage of CNN in this task is the speed. When dealing with the same size of data, CNN completed the task in 10 minutes(2 epochs) while it usually takes bidirectional-LSTM 2 hours to run. And yet, the performance of CNN is very close to that of Bidirectional-LSTM with significantly less time.

4.2.3 Bidirectional LSTM

We trained a Bidirectional-LSTM without attention mechanism with the text sequence. The embedding layer is initially weighted with the pre-trained GloVe co-occurrence matrix. The dropout rate is both 0.3 for each hidden layer. Adding dropout effectively prevents the network from over-fitting [13]. Next, we concatenated the outputs of Bidirectional-LSTM with 6 meta features. We then added two fully connected layer to the model. The architecture of the model is shown on Figure 4. In addition, each layer is activated by ReLu except for the output layer, where we used sigmoid activation.

In addition, we tried and replaced LSTM, GRU with the Bidirectional-LSTM, as they did not achieve the result as good as that of Bidirectional-LSTM [10]. We also tried to add attention mechanism to the bidirectional-LSTM, it turns out the attention helped to locate which words are highly weighted as important, but not as helpful in improving the accuracy of the model.

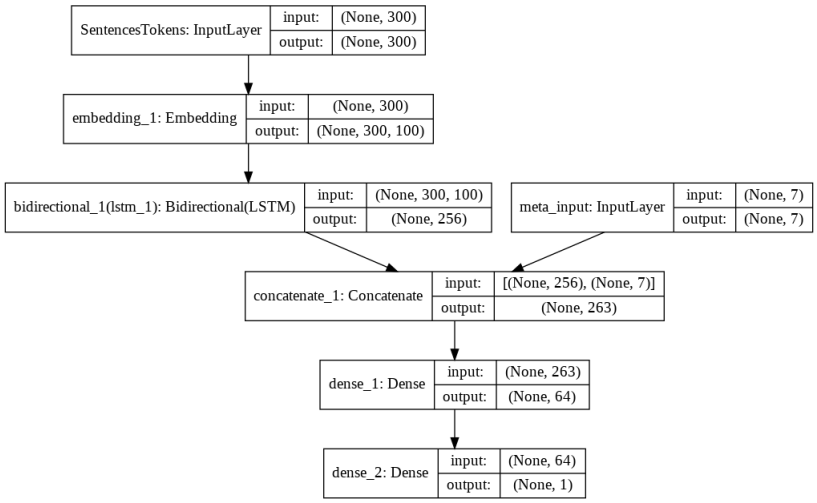

Figure 4: bidir-LSTM Model architecture

5 Result

To evaluate prediction performance for the methods, we randomly splitted the dataset into 56% training, 24% validation and 20% testing data and calculated AUC (Area Under ROC Curve), prediction accuracy (with cutoff of output at 0.5) and cross entropy loss on the testing data for each method. The results for predicting the binary label of usefulness with cutoffs at 1 vote and 9 votes are summarized in Table 2 and Table 3 respectively.

Table 2: Prediction results for the binary outcome of > 1 votes/=0 votes

>1/=0	Epoch	Dev			Test		
		AUC	Accuracy	Loss	AUC	Accuracy	Loss
Unigram and Bigram text features	2	0.739	0.676	0.604	0.722	0.688	0.594
Unigram and Bigram text features+ Meta features	2	0.810	0.733	0.533	0.802	0.735	0.554
BidirLSTM + GloVe + Meta features	2	0.811	0.723	0.550	0.808	0.758	0.451
BidirLSTM+GloVe + Meta features + Attention	2	0.785	0.719	0.552	0.793	0.734	0.462
BidirGRU + GloVe + Meta features	2	0.796	0.723	0.550	0.790	0.738	0.549
CNN+GloVe + Meta features	2	0.796	0.714	0.566	0.794	0.657	0.689

Table 3: Prediction results for the binary outcome of > 9 votes/= 0 vote

>9/=0	Epoch	Dev			Test		
		AUC	Accuracy	Loss	AUC	Accuracy	Loss
Unigram + Bigram text features	2	0.888	0.808	0.427	0.846	0.815	0.432
Unigram and Bigram text features + Meta features	2	0.958	0.895	0.297	0.945	0.845	0.486
BidirLSTM + GloVe + Meta features	2	0.928	0.869	0.325	0.932	0.947	0.303
BidirLSTM + GloVe	5	0.934	0.881	0.378	0.821	0.821	0.416
CNN + GloVe + Meta features	2	0.941	0.876	0.306	0.937	0.821	0.450

For binary outcome with cutoff at 1 (Table 2), Bidir-LSTM + GloVe + Meta features model achieves the highest test AUC (0.808) and the highest Accuracy (0.758) among the models, followed by Unigram and Bigram text features+Meta Features with AUC(0.802) and accuracy (0.735).

For the binary outcome with cutoff at 9 (Table 3), it turned out that the model of Unigram and Bigram text features+Meta Features give the decent performance with an AUC (0.945) and accuracy (0.845). The CNN + GloVe + Meta features also achieved great AUC (0.937) with accuracy (0.821).

Also, the Bidir-LSTM + GloVe + Meta features achieved best test accuracy at 0.947, with test AUC of 0.932 and loss of 0.303.

For RNN models, GRU and LSTM did not have significant difference on performance, while LSTM was slightly better by three metrics. Bidirectional-LSTM model with attention was not as good as that without attention in this specific classification task, and the running time was longer. Due to time limitation, we did not test the model with attention on the data with 9 cutoff.

We also found that including meta features could significantly improve model performance according to all evaluation metrics (AUC, accuracy, and cross-entropy loss) used in this study. And our models did not over fit the training data since prediction results on development and testing datasets do not differ too much.

6 Discussion

In this project, we explored multiple neural network models to predict useful reviews, and compared model performance using three evaluation metrics. The Bag-of-Words (bigrams and unigrams) model treats word frequency as text feature, while CNN and Bidir-LSTM models with embedding pay more attention to the co-occurrence of words in a context by initializing the weights of embedding with the pre-trained GloVe weight matrix. They have different focuses. While in some cases the AUC of Bigram model turns out to be higher than that of the both CNN and Bidir-LSTM model, the accuracy of Bidir-LSTM standed out among all the models. Based on all of the models we have explored, we believe that an ensemble model might have better performance.

It is worth mentioning that a binary classification of usefulness might not be sufficient to recommend reviews for the users. We could further divide the classes and test the results or build models for predicting the counts of usefulness votes in the future.

For future work, we can also tune all the hyper-parameters to further improve the model's performance and ensemble the outputs of various models. In addition, we can exploit the attention signals to find which words, bag-of words and meta features are important, so that we can have a better understanding on why the Bigram model have such great performance.

References

[1] Yelp dataset challenge.

[2] Ben Isaacs, Xavier Mignot, and Maxwell Siegelman. Predicting usefulness of yelp reviews. 2012.

[3] James Huang, Stephanie Rogers, and Eunkwang Joo. Improving restaurants by extracting subtopics from yelp reviews. *iConference 2014 (Social Media Expo)*, 2014.

[4] Mingming Fan and Maryam Khademi. Predicting a business star in yelp from its reviews text alone. *arXiv preprint arXiv:1401.0864*, 2014.

[5] Yoav Goldberg. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 10(1):75, 2017.

[6] Kim and Yoon. Convolutional neural networks for sentence classification. *A Determination of the Hubble Constant from Cepheid Distances and a Model of the Local Peculiar Velocity Field*, Sep 2014.

[7] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[8] Wikipedia contributors. Recurrent neural network — Wikipedia, the free encyclopedia, 2018. [Online; accessed 14-December-2018].

[9] Zachary C Lipton, John Berkowitz, and Charles Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[10] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[12] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[13] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

A Appendix

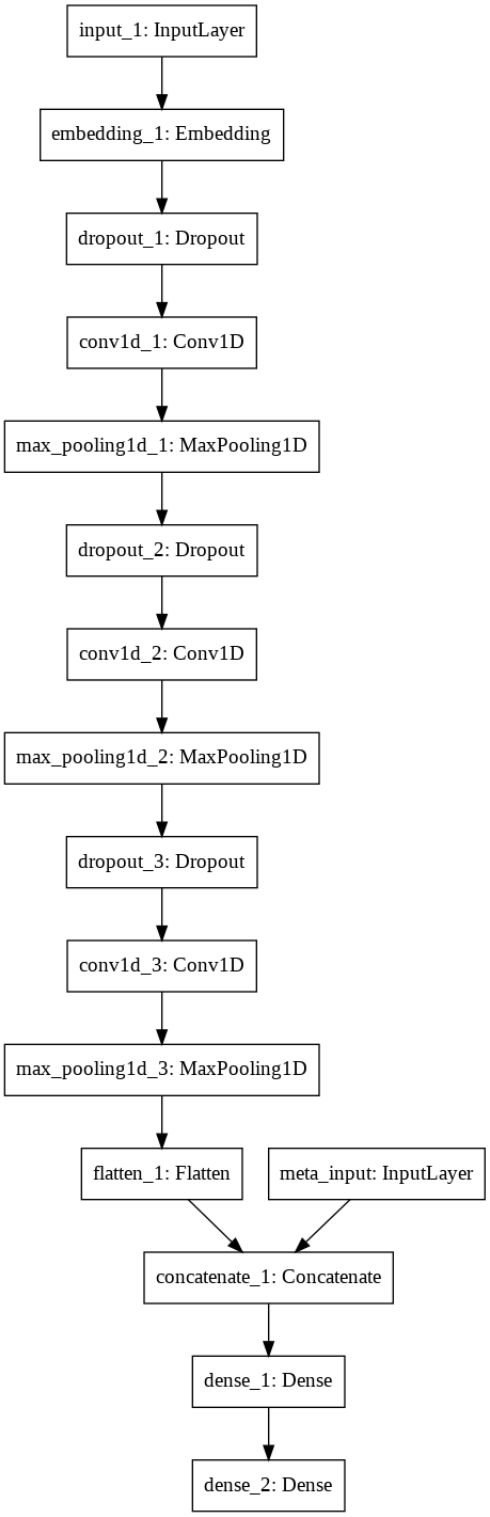

Figure 5: CNN Model Structure