Yelp Review Rating Prediction with Stacked LSTMs and Data Augmentation

Soham Kale [*1] **Hubert Luo** [*2] **Allison Yeh** [*1] **Austen Zhu** [*1]

Abstract

Various NLP architectures including stacked LSTMs, ensemble transformers, and logistic regression were applied in conjunction with data augmentation techniques to effectively predict Yelp review ratings.

1. Introduction

Sequence classification is a task in NLP that has applications to a wide variety of problems. In this paper, we will explore one such application using the Yelp review dataset released by Yelp, which consists of over 500 thousand crowd-sourced text reviews of various businesses. Using this dataset, we will develop a sequence classification model that will predict the star rating given the corresponding review text. The star rating is an integer from 1 to 5 inclusive, while the review text is a sequence of sentences written by the user to elaborate on their experience at that business establishment.

This problem is important to stakeholders in the Yelp community, specifically businesses such as restaurants that rely on accurate feedback from customers to evaluate its performance and develop improvements. This paper's results and findings may be useful in extensions to other crowd-sourced review platforms without a visible star rating. For example, a business may collect feedback informally in real time from its customers, and a summarization of the sentiment of that feedback would be useful in evaluating large-scale feedback from multiple users quickly and efficiently. In addition, more accurate prediction systems would also enable more meaningful cross-platform comparisons of the true sentiment behind different reviews, for example how a review on one platform such as Yelp compares with a review on another platform such as Google where the visible ratings may differ in meaning.

The final model uses the Mean Absolute Error (MAE) and Accuracy to evaluate performance on hold-out validation data. A baseline model of logistic regression and ensemble transformers were also used. We will describe our implementation of the final model, including data augmentation, pre-processing, and a stacked ensemble LSTM architecture.

[*]Equal contribution [1]Department of Computer Science, University of California, Berkeley [2]Department of Statistics, University of California, Berkeley.

2. Approach

2.1. Data Augmentation

The Yelp review dataset released by Yelp consisted of 533,581 reviews crowd-sourced from different users on the platform. Each review consisted of a review text and a star rating, an integer from 1 to 5 inclusive.

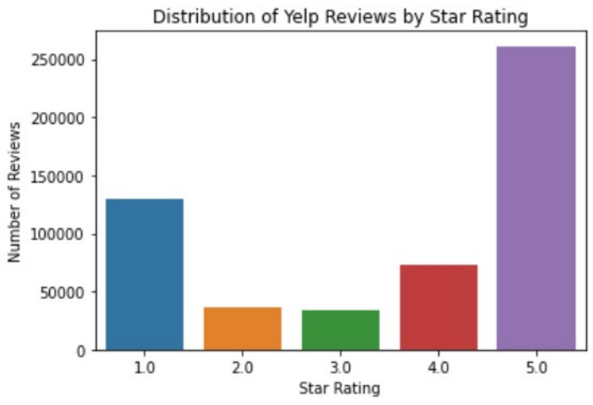

Figure 1. Distribution Yelp review star ratings in the original training dataset of 533,581 reviews

As seen in Figure 1, the distribution of Yelp reviews by star rating is heavily skewed as there were many 5-star and 1-star reviews, with much fewer 2, 3, and 4-star reviews. As a result, data augmentation techniques were employed to generate new reviews for the underrepresented star ratings based on given reviews with slight perturbations. A literature review was first conducted on previous data augmentation techniques used (Xie et al., 2019) specifically in training NLP models for text classification (Wei & Zou, 2019). Data augmentation techniques used in this paper for training the model included random deletion, random swap, and synonym replacement.

The first data augmentation method used was random deletion, which involved removing at random a certain proportion of words from the text of a review. Secondly, random swap was used, which meant switching the locations of pairs of words at random from the review text. Finally, synonym replacement was used to augment the data by replacing at random certain words in the review text with a synonym of that word. The synonym that was found was based on not only the word itself, but also the word's context and part-of-speech tag. Using these three data augmentation techniques, an additional 1.6 million reviews were generated and employed as part of the training pipeline, especially for the purpose of balancing the class labels for the dataset and achieving a more even proportion of reviews of each star rating.

2.2. Preprocessing

We used the off-the-shelf Yelp dataset. In our final preprocessing pipeline, we ended up dropping N/A values, stemming using the Snowball stemmer, truncating/padding our text sequences to 300 words in length, lemmatizing it using a trained Keras Lemmatizer, and finally embedding them into the GloVe-FastText representation.

We iterated on a number of preprocessing steps as well. We tried replacing stemming with just a trained lemmatization mechanism, to get more natural language tokens; however, it seemed to make no difference over standard stemming so we kept the former. We also tried normalizing the text, by removing special characters, lowercasing words, and removing extraneous conjunctions. However, we found that this ended up removing a lot of inherent meaning in the text: "yay" in a Yelp review has a fundamentally different intent than "YAY!!!!!" and normalization would remove the emotive aspect of the word.

2.3. Logistic Regression

A multinomial logistic regression model with a softmax layer, cross entropy loss, and L2 regularization was trained on the dataset with stop words removed. Some feature extraction methods we explored were unigrams and bigrams. Using bigrams in addition to unigrams allowed us to model the relationship between two words. For example, "not delicious" would be more helpful as a feature than the two words individually. Additionally, we applied TF-IDF (term frequency - inverse document frequency) weighting to the feature matrix. This technique puts less weight on common features and more weight on rare features because rare words act as better distinguishers between reviews.

The model was tuned on several hyper-parameters, including batch size, learning rate, number of features, number of training steps, and regularization rate. The training set was constructed from a random sample of 640,000 reviews from the Yelp dataset. The validation performance was measured on a set of 50,000 reviews separate from the training set. Table 1 shows the performance on a select few models that were trained over 5,000 training steps with the top 10,000 most frequent features.

Table 1. Validation MAE and accuracies for selected logistical regression models after 5,000 training steps with the top 10,000 most frequent features.

Model	Batch Size	Learn Rate Val MAE	Reg Rate Val Acc
Cased	128	1e-4	0.03
		1.814	0.214
Uncased	128	1e-4	0.03
		1.802	0.208
Uncased	256	1e-4	0
		1.918	0.196

2.4. Transformers

A standard 12-layer BERT model was used as the base transformer model, trained on lower-case English text with 768 hidden units, 12 heads, and a total of 110 million parameters. A dropout layer, linear classification layer, and soft-argmax output layer was used to customize the base BERT model for sequence classification. The soft-argmax was chosen to make the network fully differentiable, as opposed to a non-differentiable simple argmax. An ADAM optimizer with weight decay was used in addition to a linear learning rate scheduler to decrease the learning rate after a certain number of warm-up steps.

Various models were trained on both uncased and cased text with various hyper-parameters such as learning rate, encoding dimension, and batch size. Training was performed on a subset of the reviews dataset, specifically 10,000 reviews randomly sampled from the overall dataset. The validation dataset also consisted of 10,000 reviews with no overlapping reviews. Relatively similar results were observed on the training and validation datasets, demonstrating the transformer models were able to be generalizable despite being trained on only a relatively small subset of the data. Table 2 shows the results of selected transformer models trained after 10 epochs.

Previous work (Xu et al., 2019) demonstrated promise specifically in using ensemble transformer models to boost performance. Two different ensembling methods were applied to the most promising models trained above, max and average ensembling. Max ensembling was performed by taking the maximum probability of the star rating prediction being each class for a collection of transformer models.

Table 2. MAE and accuracies for selected transformer models with uncased BERT base, batch size of 16, and after 10 epochs.

	DIM	LEARN RATE	TRAIN MAE VAL MAE	TRAIN ACC VAL ACC
1	128	1E-5	0.205	0.831
			0.371	0.746
2	128	5E-6	0.358	0.733
			0.430	0.720
3	72	1E-4	1.383	0.499
			1.438	0.490

Table 3. Validation MAE and accuracies for selected ensemble transformer models.

MODELS	ENSEMBLE METHOD	VAL MAE	VAL ACC
1+2	AVERAGE	0.387	0.694
1+2	MAX	0.402	0.735

On the other hand, average ensembling was performed by taking the average of predicted star ratings, rounded to the nearest integer. Results are presented in Table 3 for a selected number of ensemble transformer models.

Ensembling the transformers did not demonstrate noticeable improvement on the best transformer model, likely due to the similar data used to train the best models. Future work would involve training seperate transformer models for different parts of the review text, for example the beginning and ending of a review.

2.5. LSTMs with Second Level Learning

The architecture used for the LSTMs was:

- A pretrained embedding, either GloVe or FastText
- Spatial Dropout, at 0.5 during training time
- A Bi-directional LSTM with output size 40
- A Bi-directional GRU with output size 40
- Two pooling layers, concatenated, utilizing the output of the previous layer
  - An average pooling layer
  - A max pooling layer
- A fully connected layer with sigmoid activation

All reviews (including augmented reviews) were first stemmed using NLTK's snowball stemmer, using the 20000 most important features. The model was then trained with a batch size of 1024 for 20 epochs.

A couple variations were then tested, with different max-review lengths, and unaugmented and augmented datasets. Augmented datasets added only reviews with 2, 3, and 4 stars, to the training set only. The final augmented dataset had the distribution demonstrated in Figure 2.

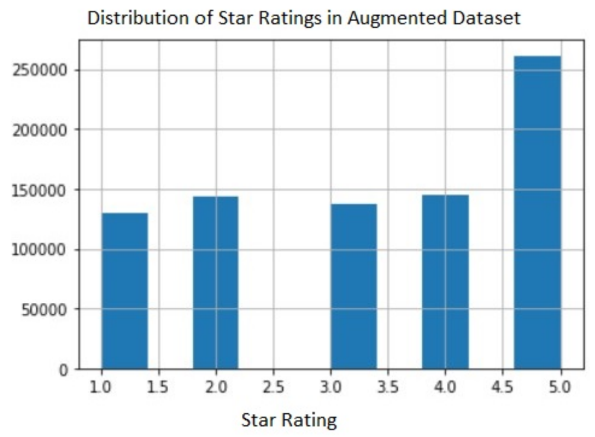

Figure 2. Distribution of Yelp review star ratings in the final augmented dataset

The final decision to use 300 words of each review was influenced by looking at the distribution of review lengths. Since a majority of reviews fell under 200 words, it was first chosen. Models using 200 and 300 words were trained on non-augmented datasets, with 300 word models being chosen for higher accuracies. This is demonstrated in Figure 3 and Table 5 below.

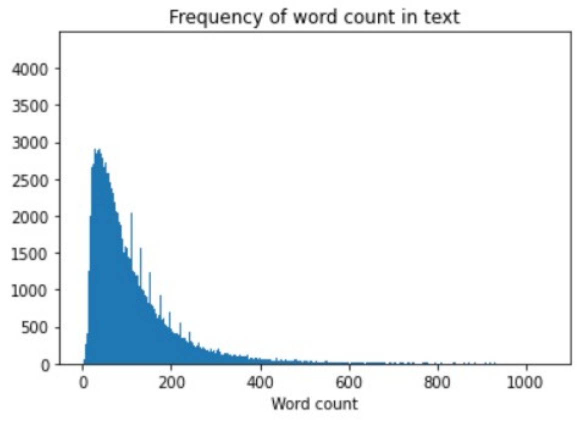

Figure 3. Frequency of word counts in review text

Table 4. MAE and accuracies for selected LSTM models with and without Data Augmentation (DA). In the model names, "F" refers to a model trained on the first number of words in a review and "L" refers to the last number of words in a review.

MODEL	DA	TRAIN MAE VAL MAE	TRAIN ACC VAL ACC
GLOVE-F200	NO	0.179 0.224	0.803 0.763
FASTTEXT-F200	NO	0.168 0.198	0.814 0.784
GLOVE-F300	YES	0.151 0.198	0.839 0.794
FASTTEXT-F300	YES	0.136 0.199	0.858 0.796
GLOVE-L50	YES	0.167 0.238	0.820 0.755
FASTTEXT-L50	YES	0.182 0.231	0.802 0.749

3. Results

The dataset used to generate the results was the augmented Yelp review dataset, based on the original Yelp review dataset of 533,581 reviews. Data augmentation was done to generate additional reviews - refer to Section 2.1 for specific implementation details. Only augmented reviews corresponding with 2, 3, and 4-star reviews were added to the original dataset to create the augmented dataset. This was done to achieve better class balance, as there were fewer 2, 3, and 4-star reviews in the original dataset.

The augmented review dataset was then split into training and validation datasets with an 80-20 split, i.e., 80% of the reviews in the augmented datasets were used for training the final model. Testing was done on the external challenge datasets.

Results for the baseline logistic regression and transformer models are shown in the previous sections in Tables 1, 2, and 3. The results for individual LSTM models, both with and without data augmentation, are also shown in Table 4.

The individual LSTM models outperformed the baseline models, with the highest validation accuracy occurring for the FastText Model trained on the first 300 words. The GloVe model trained on the first 300 words achieved the lowest validation MAE. Generally, models with data augmentation outperformed models without data augmentation and models trained on more words outperformed models trained on fewer.

Multiple text classification tasks have seen success by ensembling their models. Second level learning was attempted with the output of all models with data augmentation using XGBoost. The results for the final ensemble model are shown in Table 5.

Table 5. MAE and accuracies for final ensemble LSTM model trained on dataset with data augmentation

MODEL	TRAIN MAE VAL MAE	TRAIN ACC VAL ACC
ALL AUG. MODELS	0.095 0.193	0.905 0.807

A final validation MAE of 0.193 and validation accuracy of 0.807 was observed for the ensemble model, which outperformed all the individual models, both with and without data augmentation.

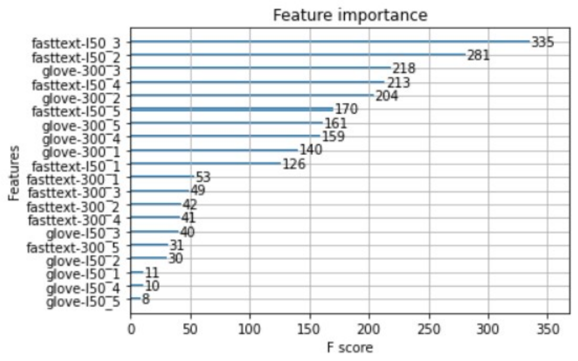

Figure 4. Distribution of feature importance scores for word subsets of reviews embedded with GloVe or FastText. For example, "fasttext-l50_3" refers to the last 50 words of three star reviews embedded with FastText.

As mentioned previously, the best performing model was an ensemble of all the individual LSTM models with data augmentation. We experimented with using different subsets of words from each review. Something we found surprising was that the last 50 words of reviews had high feature importance. According to Figure 4 above, the last 50 words of three and two star reviews had particularly high feature importance scores. Among our individual LSTM models however, we actually achieved the best performance when we used the first 300 words of each review.

4. Tools

The nltk package was used for data augmentation in order to tokenize, remove stopwords, and apply part-of-speech tagging. WordNet in conjunction with nltk was used for finding synonyms during the synonym replacement aspect of data augmentation. For the logistic regression model, tensorflow was used. For the transformer model, packages used included pytorch and transformers, in order to access pre-trained BERT tokenizers and transformers. For the Stacked LSTMs, tensorflow was used instead. XgBoost was then used for second level learning. These specific packages were employed due to previous experience using them, their extensive documentation, and relative ease of use.

References

Wei, J. and Zou, K. Eda: Easy data augmentation techniques for boosting performance on text classification tasks, 2019.

Xie, Q., Dai, Z., Hovy, E., Luong, M.-T., and Le, Q. V. Unsupervised data augmentation for consistency training, 2019.

Xu, C., Barth, S., and Solis, Z. Applying ensembling methods to bert to boost model performance, 2019.