Fake Review Detection on Yelp Restaurant Data

Ganapathy Sankararaman*(ganasank)*, Khaled Aounallah*(kaounall)*, Ferhat Turker Celepcikay*(turker)*

Abstract:

Online product reviews are a fundamental part of the decision making process for customers in e-commerce. These reviews can be classified as positive/negative based on the way they describe the product/venue. They are very influential in making a product successful/unsuccessful by leading people to choose the wrong product. Our project tackles the problem of spam/fake reviews by developing a model, which could classify a given review as either fake or genuine, thereby helping to make more meaningful review information available to the customers. We worked with 6 different models - Multinomial Naive Bayes, Logistic Regression, Neural Networks, CNNs, Gradient Tree Boosting and BER. BERT has the highest accuracy of 75%.

Introduction:

Product reviews are key in determining the reputation of the service offerers, and would reflect in the sales achieved by the business. However, since people can freely write their own contents, their opinions might not always be accurate and unbiased and can mislead other customers in their choices and decisions. Implications of such fake reviews can affect both the customer and the servicers and can pose a significant challenge in the e-commerce industry. Hence, it is necessary to identify fake reviews and ensure the integrity of the reviewing process. We developed 6 machine learning models, which take in the review content, along with user id, product id, rating and the review date as inputs, and classifies the review as either fake/real.

Related work:

Numerous attempts have been made in order to characterize reviews as fake/real, a problem that has proven to be a difficult task given how hard it is to establish a ground truth for the classification of data and how expensive it can be to manually label large data sets. In previous work, a variety of approaches were explored ranging from linguistic approaches where language patterns among opinions are analyzed for psycholinguistic tools[6], to graph-based methods leveraging state-of-the-art machine learning techniques on generated review-based and reviewer-based generated features from the content[7]. Additional foundational work was performed through the introduction of Convolutional Neural Network[3], where the application of CNNs allowed to improve sentence classification tasks. Most recently, research scientists at Google came up with a new language model - Bidirectional Encoder Representations from Transformers (BERT)[5], which has been considered to be the best so far in many NLP tasks.

In our project, we capitalize on the considerable progress made in spotting opinion spam to propose a variety of detection methods that combines text analysis and feature engineering, and machine learning and neural network classification techniques.

Dataset and Features:

I. Dataset:

In our work, we study New York City Restaurants review data from the dataset provided by the author of “Collective Opinion Spam Detection: Bridging Review Networks and Metadata”[1]. The data is organized by the date of the review and is associated with a Yelp user ID, and a Yelp product ID. Each review is also associated with a discrete rating value ranging between 1 and 5. The dataset also has labels generated from Yelp’s filtering algorithm.

The training dataset is taken from year 2011 with 5562 fake and real reviews. For the validation dataset, we randomly sampled 1000 fake and real reviews from the year 2012. The testing dataset consists of 5000 fake and real reviews randomly sampled from year 2013.

II. Features:

The reviews were cleaned with a refinement process to reduce the size of the data and eliminate any irrelevant information. Our preprocessing includes the removal of punctuation and performing word-level, character-level tokenization and n-gram based models to optimize for the choice of our training features. Our training dictionaries were constructed using snowball stemming and term frequency–inverse document frequency was used to consider the relevance of certain words than their frequency. For Neural Networks, GloVe was used for

vector representation of words. For logistic regression, count vectorizer was used.

Additional features were also incorporated in our training process including the other four provided raw features (user id, product id, rating, date of review) and 16 derived features obtained through visualizing differences between fake and real data. These added features can be divided into three buckets:

- 4 Reviewer-Centric Features: no. of reviews given by user, the average rating given by user, Average no. of words in reviews by user, Max reviews given by user in a day,
- 4 Product-Centric Features: no of reviews for restaurant, average rating for restaurant, average no. of words for restaurant review, max reviews received by restaurant in a day,
- 8 Review Centric Features: Character count, word count, word density, punctuation count, upper case word count, top 1000 unigrams and bigrams that occur more frequently in fake reviews than real reviews, top 1000 unigrams and bigrams that occur more frequently in real reviews than fake reviews.

The derived features allow for more information about the fake reviewer and the product receiving the fake review. Our primary goal is to achieve the classification with just the review and look at how the metadata helps us in performing better.

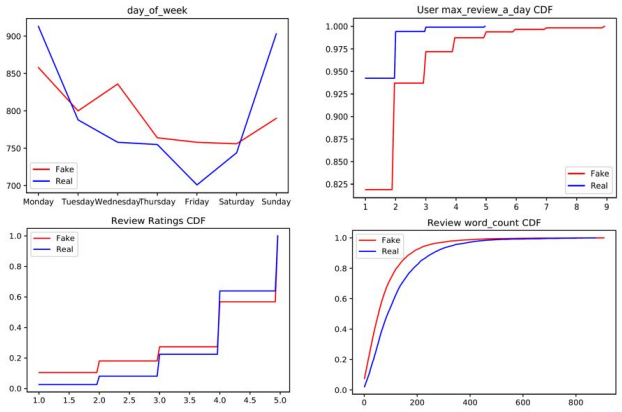

Fig 1: Data Visualization

We observed that fake reviewers tend to write more on Tuesday-Friday, while most real reviews are written on Sundays and Mondays. Also, most real reviewers wrote max one review a day, whereas fake reviewers may generate as high as 9 reviews a day. Moreover, fake reviewers tend to give more 1s and 5s ratings compared to real reviewers. Finally, fake reviews are shorter in length.

Methods:

- **Multinomial Naive Bayes**: is a generative model. The set of words in the text are encoded into a feature vector, which is called the vocabulary. This model has a very strong assumption, which is x_i's are conditionally independent given *y.* This method is known to perform well on text-based tasks.
- **Logistic Regression**: using the count vector of words, the model minimizes loss to achieve linear separation between fake and real reviews. Used mainly for binary classification. It has the property of being well calibrated.
- **Gradient Boosting Tree**: An ensemble learning techniques, combining a large number of decision trees to build the final model that will represent a forest of decision trees. Each non-leaf node in a tree represents a splitting of features, each branch represents the flow of data based on the split, and each leaf represents a classification. A tuning of the model hyperparameters was performed to determine optimal values for training. Through optimizing with respect to the AUC scores, we choose values for number of estimators, maximum features, learning rate, and maximum tree depth.
- **Neural Network**: 6 hidden layers - 3 fully connected layers for reviews and 3 fully connected layers for metadata. All hidden layers use ReLu activation and use softmax for classification. All hidden layers are L2 regularized and there's a 50% dropout rate at before the output layer. Categorical cross entropy loss is minimized. The top 1000 unigrams/bigrams which occur with the most TF-IDF between fake and real reviews were chosen as input to the Neural Network for the reviews.
- **CNN**: 128 convolution 2D filters for 5 different filter sizes (1-5), and have ReLu activation, followed by max pooling. The convolution layer is L2 regularized and has normal initialization for the kernels. This is for the reviews with GloVe vector representation. In addition to this, there are 3 hidden layers for the metadata (L2 Regularized), all of which have ReLu activation. The output layer uses softmax and there is a 50% dropout

before the output. Categorical Cross entropy is minimized in the model.

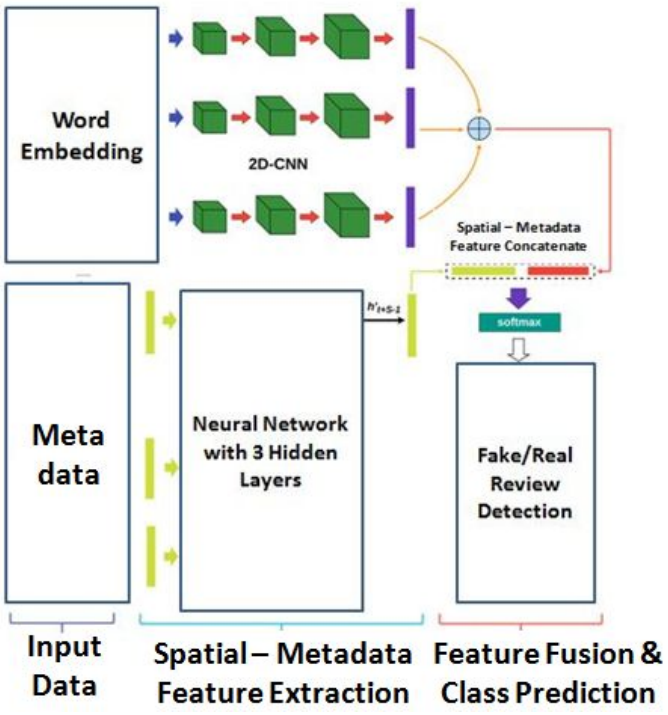

Fig 2: CNN Architecture

- **BERT**: BERT became very popular in the machine learning community after presenting superior results in a wide variety of NLP tasks. Its key difference with previous efforts is to look at a text sequence either from left to right or combined left-to-right and right-to-left training. By using a novel technique - Masked Language Modelling (MLM), bidirectional training in models is achieved.

Experiments:

Neural Nets and CNNs: The hyperparameters - L2 regularization values, batch size, kernel initializer and optimizer (with different learning rates) were chosen from a set of 4 different values chosen from various papers and sources for grid search. Validation dataset accuracy was used to choose the best parameters. With these, the model was trained for 40 epochs, while using checkpoints to keep track of optimum training weights. These were then chosen for the final predictions.

GloVe and wiki news word vectors were used. GloVe gave best results. The model was also trained and tested without these vectors. In both the above iterations, the weights for these words were further set to trainable/not trainable to see the effect. In all the cases, the results were comparable. This is consistent with how all the text based models perform comparably.

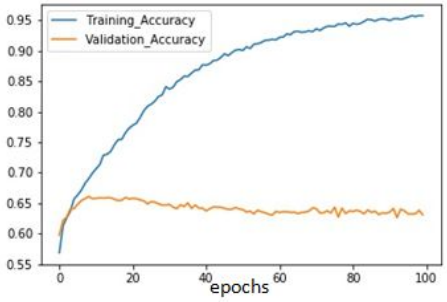

Fig 3: Train and Test Accuracy during CNN Training

Also, once the text based NNs and CNNs were trained, the metadata was fed to a separate neural network architecture, after which the output of both the text based and metadata based NN were concatenated. This helped in improving the performance of the model.

Batch size - 64, learning rate = 1e-3, dropout - 0.5, normal initializer, Adam optimizer, l2 reg - 0.005(CNN), 0.01(NN).

Gradient Tree Boosting: In our model, we choose to perform different experiments on the relevant hyper parameters - learning rate, number of estimators, maximum depth of the tree and the maximum features parameter to represent the number of features considered when looking for the best split. Initially, we look into the performance of the baseline model using default parameters using AUC as an evaluation metric (it's good for binary classification). Some of the results are captured in the following plots and we end up choosing the parameters that show the best performance on the validation set, while trying to minimize any overfitting effects in our case.

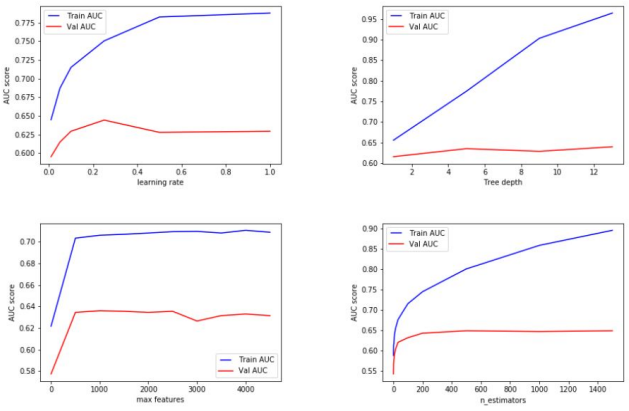

Fig 4: Gradient Tree Boosting Hyperparameter selection

Logistic Regression and Multinomial Naive Bayes: Using count vectorizer in the training with the default setting of hyperparameters resulted in high variance. Upon literature search we took two steps to mitigate high variance, namely feature reduction (selection) and hyper parameter tuning. We first used Principal Component Analysis (PCA) to reduce the number of features to 300. Then, we tuned hyperparameters by running a grid search. Later, as an alternative to PCA, we tried Latent Dirichlet Allocation(LDA)[8] model. The performance was not as good as PCA. Finally, to assure overfitting is mitigated, we ran 5-fold cross-validation on all datasets. We also explored Adaboost. It did not produce any better results.

BERT: Hyperparameters including 'keep_prob', the batch size and the number of train Epochs were tuned. 'Keep_prob' is the dropout hyperparameter which helps prevent overfitting. 'Keep_prob' was set to 0.7. Mini batch size is 16. Learning rate is 2e-5.

Results and Discussion:

We use accuracy, recall, precision and F1 score. Accuracy alone is not a good indicator of the model's performance. Precision gives the information of how accurate the positive predictions are. On the other hand, recall gives the percentage of true labels predicted correctly. The F1 score combines both of these and give a single value. The AUROC gives area under the specificity-sensitivity curve. This was used for fine tuning models as well as this is a good indicator of model performance. For binary classification, AUROC is generally sufficient. So, we didn't use AUPRC.

Table 1: Performance Measures

	Accuracy		AUROC		F1 Score			
Model	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**PR**	**RC**
NN	0.73	0.69	0.80	0.75	0.75	0.72	0.65	0.82
CNN	0.73	0.70	0.80	0.76	0.75	0.73	0.67	0.81
GB	0.74	0.66	0.83	0.71	0.74	0.68	0.63	0.71
BERT	0.94	0.75	0.94	0.83	0.94	0.77	0.71	0.83
LR	0.69	0.65	0.76	0.72	0.71	0.69	0.63	0.75
MNB	0.71	0.65	0.77	0.71	0.72	0.68	0.63	0.74

PR - Precision for test, RC - Recall for test

For Error Analysis, we individually (3 members) tried classifying 20 misclassified reviews by worse margin from BERT. We took the classification that was most chosen by the three of us, and compared it with BERT's prediction. 50% of what BERT predicted was correct. Thus, there is a good chance that YELP's filtering algorithm might have misclassified the reviews. This could be the reason for the bias error.

Also, with bootstrapping of test set with sample size of 2000, we found that the standard error is a bit more than 1%. So this could have contributed to the lower accuracy (due to variance) in testing dataset as well.

From the Calibration curve, we can see that except BERT, all the other models are well calibrated. It shows that well calibrated models need not have high accuracy.Also, from the AUROC, we can see that BERT has the largest area under the curve, followed by CNNs and NNs. This explains the performance of the models in terms of accuracy as well.

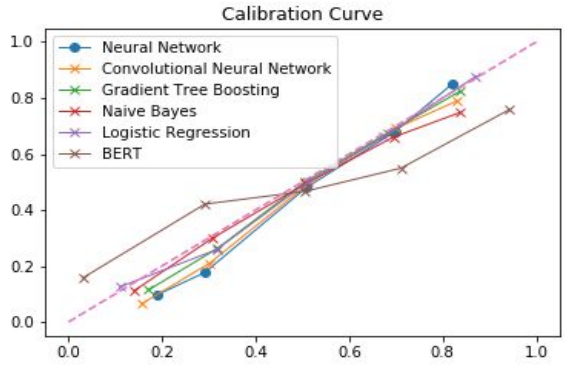

Fig 5: Calibration Curve

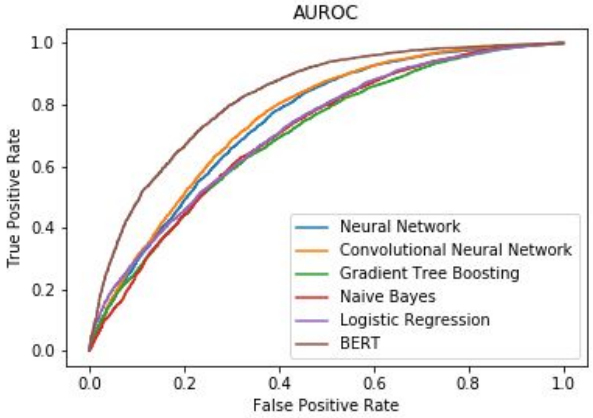

Fig 6: Sensitivity - Specificity Curve - ROC

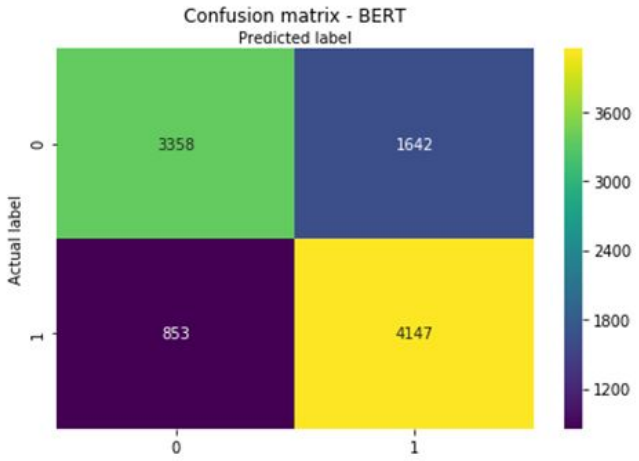

Fig 7: Confusion Matrix - BERT

From the confusion matrix for BERT (and Precision/Recall values for all models), it can be seen that one third of the real reviews are misclassified as fake reviews. During Error Analysis, we observed that these were among the reviews which contributed to the bias in the dataset.

Conclusion:

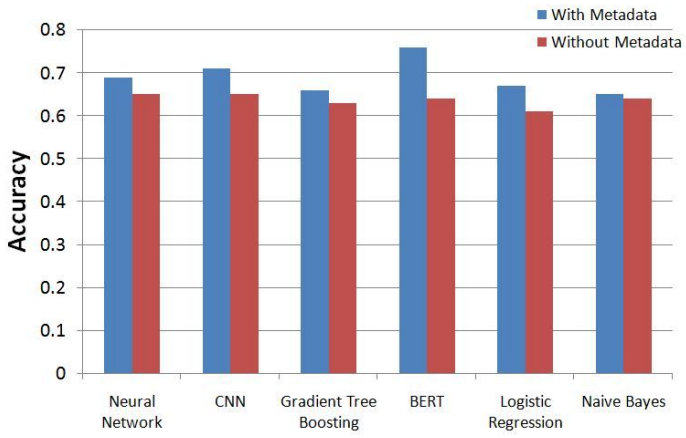

Fig 8: Accuracy - with and without Metadata

From the results, we can see that all the models have about 65% accuracy when just the reviews were used (Without Metadata). But once the metadata were used, the accuracy got boosted in all the models. BERT got an accuracy boost of about 10%. This explains that Yelp's spam filter works not just based on the reviews, but also based on the metadata. So, with the metadata, the model was able to predict the reviews better.

Future Work:

- Hyperparameters of CNN will still have to be fine tuned to see if we can get better accuracy.
- Bidirectional LSTM has been implemented already. But the accuracy was 65% with just the text. So, hyperparameter tuning will be critical to improve the performance. We'll be working on it.
- Hyperparameter tuning of a CNN - Bidirectional LSTM - metadata NN model has to be done to achieve the best possible accuracy.
- For the gradient boosting tree, higher computational resources would allow to perform further tunings of the hyperparameters and explore tradeoffs between learning rate and number of estimators. At higher values for the parameters, the computation gets slow and make it difficult for perform multiple performance tests.
- Hyperparameters of BERT will still have to be fine tuned in order to alleviate overfitting and achieve better accuracy. It should be calibrated by using calibration techniques such as Platt Scaling.

Contributions:

All the three of us contributed equally.

Code Listings:

https://github.com/ganasank/CS229-Project

References:

1. Shebuti Rayana, and Leman Akoglu. "Collective opinion spam detection: Bridging review networks and metadata." *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining*. ACM, 2015.
2. Hadeer Ahmed, Issa Traore, and Sherif Saad. "Detecting opinion spams and fake news using text classification." *Security and Privacy* 1.1 (2018): e9.
3. Yoon Kim. "Convolutional Neural Networks for Sentence Classification."(2014).
4. Shervin Minaee, Elhaam Azimi, Amir Ali Abdolrashidi. "Deep-Sentiment: Sentiment Analysis Using Ensemble of CNN and Bi-LSTM Models." (2019)
5. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: pre-training of deep

bidirectional transformers for language understanding.” *CoRR*, abs/1810.04805, 2018.

6. Song Feng, et al. "Distributional footprints of deceptive product reviews." *Sixth International AAAI Conference on Weblogs and Social Media*. 2012.
7. Leman Akoglu, Rishi Chandy, and Christos Faloutsos. "Opinion fraud detection in online reviews by network effects." *Seventh international AAAI conference on weblogs and social media*. 2013.
8. David M Blei, Andrew Y Ng, and Michael I Jordan. “Latent dirichlet allocation.” *Journal of Machine Learning Research*, 3(Jan):993-1022, 2003.