SI 630 Project Report

Mingliang Duanmu
duanmuml@umich.edu

1 Project Goals

With the vast enrichment of modern material life, people gain access to personalizing their lifestyle in various aspects. Eating out has become a habitual activity for a great portion of people, especially for the young and those who are enthusiastic about delicacies. With hundreds of restaurants scattering over a few miles around, it is really a puzzle to choose the most preferred one, let alone people with allodoxaphobia. Consequently, mobile applications like Yelp came out to provide abundant information including pictures, specialties and most importantly, customer reviews of each restaurant. Yelp does a good job in filtering bad restaurants and let those with good taste and service stand out. However, great chances are that some restaurant owners use malicious methods to generate fake positive reviews for themselves or evil feedback for competitors in order to attract more customers. To reduce such misleading information for customers, we decide to implement a model that detects fake restaurant reviews.

2 NLP Task Definition

The major NLP task of this project is to apply sentiment analysis to a dataset of restaurant reviews to distinguish artificial reviews that are either generated by computers or paid posters. There are several previous studies focusing on the subject, so in this project a mixed model will be developed to aggregate advantages in different approaches to achieve a higher accuracy.

The goal of the project is to:

1 Research and study on current methods of detecting fake reviews and improvements on sentiment analysis, define a combined standard for evaluating whether a review is fake or not.

2 Build a machine learning model to train on the dataset.

3 Apply the model to the test set to validate its correctness.

4 Make conclusions on the models, make comparisons for different design of the model, analyze future enhancement directions.

The input of the system is a long list of reviews along with its metadata like creation time, rating and user ID. The output will be a subset of the input which are considered artificial reviews by the system.

3 Data

The dataset for this project comes from the Yelp Open Dataset that can be downloaded from Yelp's official website. A subset of `yelp_academic_dataset_review.json` will be chosen for two reasons: the whole dataset is too massive to deal with and the focus is on restaurant reviews in English. Each review in this dataset includes the business ID, user ID, date of post and feedbacks from other users besides the content, which may serve as useful parameters assisting the text analysis.

As the raw data is particularly bulky and in nested JSON format, it would be time-wasting to do data preprocessing on a personal computer. Luckily, we find the YelpZip dataset[1](Rayana and Akoglu, 2015) online and gain access to it. The YelpZip dataset is cleaned and first used by Rayana and Akoglu with reviews from four states of the United States including NJ, VT, CT, and PA. Each review has metadata including restaurant name, user id, rating and date. Also, each review is associated with a label indicating whether it is filtered by Yelp.

[1]http://odds.cs.stonybrook.edu/yelpzip-dataset/

We use Jupyter Notebook to perform a simple analysis of the dataset and some quick facts are listed below.

Keyword	Value
Reviews	608458
Restaurants	5042
Users	260239
Review Length Range (words)	[1, 5333]
Average Review Length (words)	85
Fake Reviews	80461
Fake Ratio	13.22%
Fake Review Users	60908
Fake Review Users Coverage	23.4%
Fake Review Restaurant	4334
Fake Review Restaurant Coverage	85.96%

Table 1: Basic Statistics of YelpZip Dataset

Here is an example of filtered fake review: *Drinks were bad, the hot chocolate was watered down and the latte had a burnt taste to it. The food was also poor quality, but the service was the worst part, their cashier was very rude*. We can see the sentiment is extremely negative and aggressive.

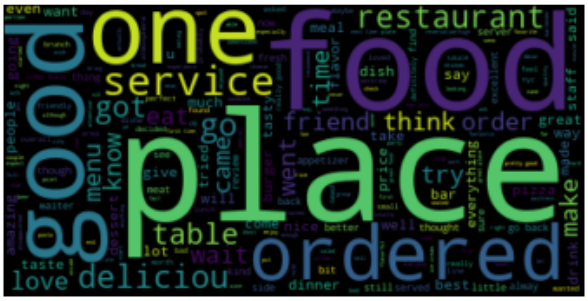

Figure 1: Word cloud for reviews

4 Related Work

- *Opinion spam and analysis*
  Based on the analysis of 5.8 million reviews and 2.14 million reviewers from Amazon, the author divide spam opinions into three types based on the large amount of duplicate reviews in the dataset. Supervised learning with manually labeled training examples is used to deal with the relationship of user ID and highly similar reviews. (Deng and Chen, 2014)

- *Using appraisal groups for sentiment analysis*
  The author explored a new method for sentiment classification based on extracting and analyzing appraisal groups like "very good" for a more precise emotion detection. Movie reviews based on this approach has an accuracy up to 90.2%. (Whitelaw et al., 2005)

- *Sentiment Analysis Based Online Restaurants Fake Reviews Hype Detection*
  The author proposed an algorithm to detect restaurant reviews from hype based on sentiment analysis, where reviews are categorized into four dimensions: taste, environment, service and overall attitude. The accuracy is of the algorithm is 74%. (Jindal and Liu, 2008)

- *An Empirical Study on Detecting Fake Reviews Using Machine Learning Techniques*
  The author study on online movie reviews by performing text classifications based on sentiment analysis to determine fake reviews. Four machine learning techniques including Naive Bayes, Support Vector Machine, K-Nearest Neighbors and Decision Tree are implemented and compared in terms of accuracy. The experimental outcome turns out that SVM performs best with an accuracy of 81.75% regardless of removal of stopwords. This training procedure proposed in this paper constitutes the basic structure of the project workflow. (Elmurngi and Gherbi, 2017)

- *What Yelp Fake Review Filter Might Be Doing?*
  The author study on the fake reviews filtered by Yelp's algorithm, conduct an analysis using the AMT model for crowdsourced fake reviews but find the high-accuracy model performs bad. It is revealed that behavioral features yielded 86% accuracy while linguistic features are not as effective. The author conclude that Yelp's filter is reliable and may be applying a behavioral based approach correlated with abnormal spamming behaviors. (Mukherjee et al., 2013)

5 Methods

The workflow of our training method resembles the scheme proposed by Elmurngi and Gherbi:
Step 1: Data preprocessing
We divide the dataset into 80% training set and 20% test set, the division should ensure the same portion of fake reviews in both sets, which we

choose 1: 1 to ensure a simple sampling. We apply some preliminary operations to better interpret the raw data as numerical values. We use `sklearn`, `nltk` and `spacy` to get word embeddings and labelling as they are among the most commonly used NLP and machine learning frameworks. At the same time, we come up with a list of stopwords to kick out useless tokens while retaining potential features of fake reviews.

Step 2: Feature selection

This is the key step where we combine different optimizations of sentiment analysis into our own model. The review features includes the length of a review, the number of words with all capital letters in a review, the number of exclamation marks used in a review, the number of subjective words (e.g. I, my, we, us...) used in a review, the rating of the restaurant associated with the review. For overall sentence feature, we have three different methods according to different models. For the machine learning model based on `sklearn`, we use a BERT based sentence embedding from `sentence_transformers`[2] to embed each review into a 768-dimension array. For the deep learning model, we try the FastText and CharNGram embedding to tokenize and embed each review to a 2d array with 300-dimension embedding for each word. The input sequence length is limited to 400 to ensure most of the reviews are covered and the data is not so bulky. To embed numerical features into the text for deep learning, we add a special token as padding token to the end of the sentence, convert the numeric values into word representing extents (e.g. short, medium, long for review length) and concatenate to the end of the review. For the deep learning model, we use one input LSTM layer, together with two linear hidden layers with a size of 128.

Step 3: Model construction

The Python package `sklearn` is a simple and efficient tool to implement different classification algorithms. We choose a set of models including: LogisticRegression, LinearSVC, DecisionTreeClassifier, RandomForestClassifier, GradientBoostingClassifier. For another model, we use a modified Pytorch model based on Arie Pratama Sutiono[3].

[2]https://www.sbert.net/

[3]Deep Learning For NLP with PyTorch and Torchtext: https://towardsdatascience.com/deep-learning-for-nlp-with-pytorch-and-torchtext-4f92d69052f

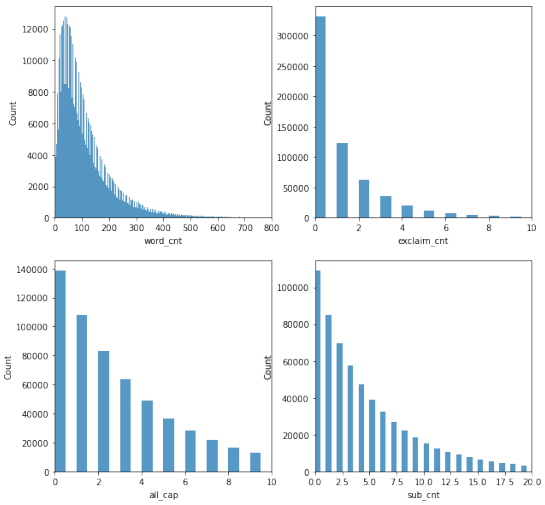

Figure 2: Basic feature distribution

Step 4: Result interpretation

We compare our predicted result on the test set to the original label to get a simple accuracy score of our model. For the machine learning model, we simply use `model.score` and `sklearn.metrics.cross_val_score`. For the deep learning model, the loss and accuary of each epoch is logged as output to supervise the convergence and performance of the model. If there is not a high increase in accuracy, we will spend some time analyzing which step goes wrong.

6 Evaluation and Results

6.1 Evaluation

The evaluation metric is simply the accuracy of detecting fake reviews in percent. Since we perform a supervised learning based on the 80% training set, a prediction and comparison on the rest 20% test set is a convenient way to validate our model performance.

After simple data processing, we are **unable to find significant differences** on distributions of metadata (user id, time, restaurant name, review length, rating) or simple sentiment scores for real and fake views, which means we cannot construct a simple baseline rule for accuracy. To achieve a better classification, we should perform a more complex strategy to add to accuracy. Therefore, we run a random classification based on probability to determine the baseline, which is about 50%.

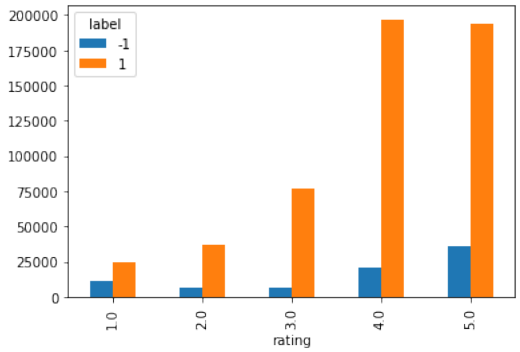

Figure 3: Rating score for real and fake reviews, no distinct distribution differences between two classes

6.2 Results

6.2.1 Machine Learning Model

Unfortunately, the pure machine learning methods do not make great improvements compared to the random 50% baseline. Therefore, it is **added as another baseline** as a trivial solution to the problem.

Classifier	Accuracy (%)	Deviation (%)
LogisticRegression	55.20	0.38
LinearSVC	53.98	0.53
DecisionTreeClassifier	51.34	0.47
RandomForestClassifier	54.72	0.45
GradientBoostingClassifier	55.57	0.63

Table 2: Machine learning model cross validation accuracy

The build-in `sklearn` models with default hyper-parameters only raise the accuracy to 6% at maximum, but through fine-tuning the accuracy score may increase a little bit. However, we did not fine-tune the model because `sklearn` does not take advantage of GPU, which means it takes a long time to iterate through all the five models once (about 2 hours on my i5-10600K processor) and my Jupyter Notebook kernel will crash halfway. Therefore, simply using the build-in models is not a good method to improve accuracy.

6.2.2 Deep Learning Model

At first, the neural network contains two linear layers and one ReLU layer, the performance is slightly higher than the machine learning results, yielding an accuracy of 60%. With the help of an additional LSTM layer and removing the ReLU layer, we achieve an amazing accuracy of 87.5% with the deep learning model. We run the training for 5 epochs to reach a stable convergence and accuracy score.

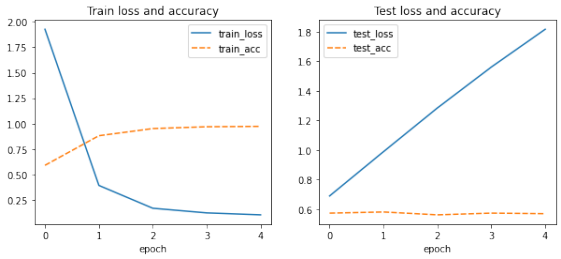

Figure 4: Deep learning model loss and accuracy without LSTM layer

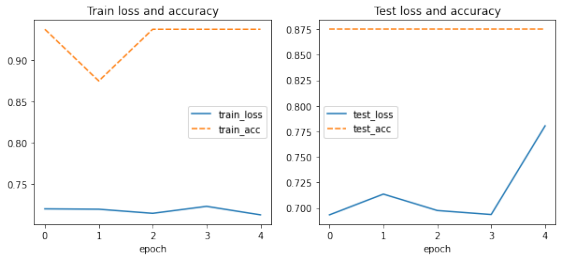

Figure 5: Deep learning model loss and accuracy with LSTM layer

A brief ablation study

After removing the metadata features from the texts, the final accuracy score suffers a slight average decrease of 2% in both machine learning and deep learning models, which means the effect of metadata is relatively small or insufficient to classification. However, the loss during training and testing experience a larger fluctuation and end up with greater values once features are removed. Therefore, more data preprocessing should be done with the meta data, or the metadata should contain more than just basic information.

7 Discussion

As this is my first NLP project, I may not gain a satisfying result combining multiple approaches and my optimization direction may be wrong. Luckily, the powerful language model has led to a good result of 87.5% beyond my expectation. Compared to the baseline of 50% and 55%, the good performance of the deep learning model is due to three major reasons: LSTM model, large embedding size and partial training data sampling. The sequential feature of LSTM enables the model to capture some sort of dependencies through time, which gives a deep understanding of fake reviews. As mentioned in the last section, the metadata and basic review features share similar distributions between real

and fake reviews, so LSTM serves as a good tool to overcome those difficulties. Also, two large hidden layer size of 256 and 128 supports more specific features, which also adds to the accuracy. When it comes to the training data selection, as fake reviews only account for 13% in the overall huge dataset, we decide to randomly sample the real reviews to the same number of fake ones to reduce the training burden. The sampled real reviews may contain less outliers that disturbs the language model. However, as moving on to the implementation of model, I find it particularly hard to replicate the method in the papers due to lack of experience in NLP field. Therefore, I use my own "dumb" methods to encode the features into text, which only brings about 2% accuracy improvement. In conclusion, my model does a good job with the help of LSTM and neural network, it can be used as the foundation of a better feature engineering and studying how to convert them into a better form as input to the training data.

8 Conclusion

In all, we build a classifier on fake reviews based on the YelpZip Dataset. First, we do data cleaning and information integration on the source files to get a list of views together with metadata. Second, we do some basic feature engineering to extract metadata and sentiment information and embeddings of the reviews. Then we build up two systems: a collection of machine learning classifiers based on `sklearn` and a deep learning model based on `pytorch`. The experimental results shows that the improvement for the machine learning model against the random baseline is only about 6%. In contrast, the deep learning model containing a LSTM layer with N-gram embedding obtains high accuracy on the testing data at a correctness of 87.5%. In conclusion, this project provides a chance for me to be fully involved in an interesting and challenging NLP project, my model finally yield a high accuracy. My experimental results demonstrates the major challenge of fake review filtering: the feature engineering from texts, so further studies may focus more on the process of text for deeper level of information extraction.

9 Other Things We Tried

To implement a simple version (due to limited knowledge, I am unable to reproduce the lexicon in the same way) of the classification method proposed in (Deng and Chen, 2014), I tried various methods including: calculating sentence similarity between keyword and reviews, calculating average word similarity between keyword and review tokens after stopword removal, constructing a bag-of-words model for each dimension and compute a weight for each review according to frequency after stopword removal. However, none of these approaches has a good effect, as manually checked by myself for a set of random samples. Such attempts are time consuming and computationally demanding, on which I wasted a lot of time.

10 What You Would Have Done Differently or Next?

In all the project result is out of my expectation, the detection of fake reviews by the deep learning model is quite efficient. Although I tried BERT based sentence embeddings from `sentence_transformers` library for the embeddings of the reviews, together with a several metadata, it is still not enough to extract features to distinguish between real and fake reviews. Therefore, a deep learning model is required to at least guarantee the accuracy improvement. However, due to limited computing resources (I only have a GPU that is good for gaming but slow for deep learning, and even Google Colab Pro did not offer me good CPUs good as my own computer for large dataset processing, worst of all, Jupyter Notebook frequently crashes on my computer when dealing with huge data), I was not able to try a wide range of methods for feature engineering for my deep learning model. If I could start some of the parts over, I may implement LDA topic analysis to classify the reviews in terms of service, taste and environment. Also, I may try fine-tuning my neural network layers for my deep learning model for even better accuracy.

References

Xiaolong Deng and Runyu Chen. 2014. Sentiment analysis based online restaurants fake reviews hype detection. In *Web Technologies and Applications*, pages 1–10, Cham. Springer International Publishing.

E. Elmurngi and A. Gherbi. 2017. An empirical study on detecting fake reviews using machine learning techniques. In *2017 Seventh International Conference on Innovative Computing Technology (INTECH)*, pages 107–114.

Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, page 219–230, New York, NY, USA. Association for Computing Machinery.

Animesh Mukherjee, V. Venkataraman, B. Liu, and N. Glance. 2013. What yelp fake review filter might be doing? *Proceedings of the 7th International Conference on Weblogs and Social Media, ICWSM 2013*, pages 409–418.

Shebuti Rayana and Leman Akoglu. 2015. Collective opinion spam detection: Bridging review networks and metadata. In *Proceeding of the 21st ACM SIGKDD international conference on Knowledge discovery and data mining, KDD?5*.

Casey Whitelaw, Navendu Garg, and Shlomo Argamon. 2005. Using appraisal groups for sentiment analysis. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, page 625–631, New York, NY, USA. Association for Computing Machinery.