Yelp Review Sentiment Analysis and Fake Review Detection

Hannah Choi, Joshua Lin, Pauline Yue

April 2021

Abstract

Fake reviews are a common problem on review-based websites, like Yelp, as they can make or break small businesses. In order to defend against issues raised by fake reviews, this research applies natural language generation and processing techniques (simple neural networks, n-gram language models, LSTMs, BERT, DistilBERT, GPT-2) to conduct both sentiment analysis and fake review detection on both real and fake reviews. Overall, our models performed comparatively better than human baseline evaluation scores for fake review detection, and our findings contribute to the growing interest in detecting AI-generated text that proliferate throughout the Internet.

Keywords: fake review detection, n-grams, GPT-2, BERT, DistilBERT, sentiment analysis, natural language processing, natural language generation

Introduction

The advancements in artificial intelligence have led to the creation of bots generating fake reviews. Here, we define fake reviews as reviews falsely written by humans or AI (but more generally by AI) to improve or tarnish a business's brand. Many small businesses rely on online reviews, such as those from Yelp, to grow and sustain their reputation. Yelp is on over 30 million unique devices. The users come from a fairly even distribution of ages– 30% range from 18-34, 36% from 35-54, and 33% 55+.[1] The presence of false negative reviews on Yelp could be detrimental to businesses.

The ultimate goal of our project is to create a model to identify fake reviews, in the hopes that this tool would improve the reliability of user-centered review platforms. In addition, we applied our NLP skills to perform sentiment analysis on each review. Due to the lack of public datasets labelling fake reviews, we created our own fake reviews and generated thousands more by using an n-gram text generator and GPT-2. We combined the public Yelp dataset and our generated fake review to train and test our classification models.

Background

Many consumers reference online review websites to help determine if they should utilize a certain business; fake review generation has increased to influence consumers' purchasing decisions. In order for consumers to make informed decisions, as well as to protect business owners from false negative reviews, the task of fake review detection has only grown in importance during the last few years. Current approaches to fake review detection

[1] "Fast Facts." Yelp, www.yelp-press.com/company/fast-facts/default.aspx.

mainly include utilizing XGBoost[2], SVM[3], and logistic regression. While we came across other novel methods in literature, such as PU-LEArning[4] which is usually used for spam detection, we decided to utilize a few deep learning methods to tackle detection of fake Yelp reviews.

The main challenge was generating realistic fake reviews, since the Yelp dataset we found most likely has fake reviews mixed in, but as the data is unlabeled we are unable to tell which are fake or real. We chose to focus on building a neural network for sentiment analysis[5] and implementing n-grams and GPT-2[6] for fake review generation to train our models on. We then trained our model on a mixture of generated fake reviews and real reviews from the Yelp dataset to test our model's accuracy. While one paper[7] utilized Amazon Mechanical Turk to generate pseudo fake reviews, we generated our own by hand in order to compare our model's ability to detect fake reviews in comparison to the human capability to detect fake reviews. In addition to neural networks, we used other language modeling methods commonly applied for fake review detection, such as n-gram[8], BERT[9], and LSTM[10].

Methods

For our data, we obtained a set of 10,000 Yelp reviews from Kaggle, related to a variety of establishments: restaurants, bars, salons, repairs, plumbing, etc. Using this diverse set of review data, we conducted both sentiment analysis and fake review generation and evaluation for our fake review detection component.

Sentiment Analysis

First, we pre-processed the text: removing html tags, white spaces, accented characters, numbers, and stop words; lower casing all words; expanding contractions; and tokenizing all remaining words. Using the pre-processed text, we ran the reviews through TextBlob to assign each review a sentiment score with a range of [-1, 1]. We then assigned labels to each review as either 1 for positive—for scores (0, 1]—or 0 for negative—for scores [-1, 0).

After labeling our data, we used the built-in keras tokenizer to create word embeddings and then divided our data into training (6,000 reviews), development (2,000 reviews), and test sets (2,000 reviews). We constructed a four-layer neural

[2]Andre Sihombing and A.C.M. Fong. "Fake Review Detection on Yelp Dataset Using Classification Techniques in Machine Learning".
[3] Zehui Wang, Yuzhu Zhang, and Tianpei Qian. "Fake Review Detection on Yelp".
[4] Huayi Li, Zhiyuan Chen, Bing Liu, Xiaokai Wei, and Jidong Shao. "Spotting Fake Reviews via Collective Positive-Unlabeled Learning".
[5] Yun Xu, Xinhui Wu, and Qinxia Wang. "Sentiment Analysis of Yelp's Ratings Based on Text Reviews".
[6] David Ifeoluwa Adelani, Haotian Mai, Fuming Fang, Huy H. Nguyen, Junichi Yamagishi, and Isao Echizen. "Generating Sentiment-Preserving Fake Online Reviews Using Neural Language Models and Their Human- and Machine-Based Detection".
[7] Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie Glance. "Fake Review Detection: Classification and Analysis of Real and Pseudo Reviews."
[8] Hadeer Ahmed, Issa Traore, and Sherif Saad. "Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques".
[9] Alex Wang and Kyunghyun Cho. "BERT has a Mouth, and It Must Speak: BERT as a Markov Random Field Language Model".
[10] Alberto Bartoli, Andrea De Lorenzo, Eric Medvet, Dennis Morello, and Fabiano Tarlao. "'Best Dinner Ever!!!': Automatic Generation of Restaurant Reviews with LSTM-RNN".

network and fitted it on our training data. Our first hidden layer contained 100 neurons, our second 50 neurons, and our third 25 neurons. We ran our model for 50 epochs using the adam optimizer and the binary cross entropy loss function, and between each hidden layer, we used a drop-out rate of 0.7 to avoid overfitting. For our hidden layers, we used the swish activation function, and for our final output layer, we used the sigmoid activation function. We used the swish activation function for our hidden layers based on pre-existing research on the relative strength of the array of activation functions, and we used the sigmoid activation function since we had a binary classification problem. Once our model finished training, we tested against our development set. We chose these parameters because after training for 50 epochs and testing other hyperparameters, we saw that the accuracy and loss would eventually plateau around 50 epochs.

After training our baseline model, we tried several steps to improve our accuracy and reduce the loss further. First, we swapped our first hidden layer with an LSTM layer with 15 neurons. We only used 15 neurons to speed up our model. Then we tried using BERT's tokenizer to process our text data to build our word embeddings. Finally, we trained both DistilBERT and BERT on our text data to obtain the feature weights for each sentence's CLS token and used those weights to train logistic regression models and our neural network model to see if any improvements could be made.[11] In addition, we increased the number of epochs to train our models in attempts to see if greater accuracy could be obtained. We used both DistilBERT and BERT, because we wanted to compare the strength of using the lighter, but faster DistilBERT compared to the slower, but more comprehensive BERT. Of note, however, is that for the models trained on the weights obtained from both DistilBERT and BERT, we had to decrease our entire dataset to just 700 reviews, since we lacked the processing power and RAM to train and test on more than 700 examples.

Fake Review Detection

Because our dataset did not include any labeled data for whether the reviews were real or fake, we used natural language generation models to produce fake reviews for our training data. First, we wrote 38 fake reviews by hand so that we could combine it with our original Yelp dataset to bootstrap additional fake reviews. To do so, we first attempted to use BERT following Wang and Cho's paper in which they used BERT's pre-trained model to generate sentences.[12] However, due to issues using applying BERT to our combined dataset of our fake and real reviews, we decided to take two different approaches.

For our first approach, we built an n-gram model as mentioned above. Pulling from our original Yelp dataset, we used an n-gram text generator (using tri-grams, 5-grams, and 7-grams) to create 1336 fake reviews. Then, we used GPT-2 and trained it on our fake and real dataset to create 3120 reviews. After building additional reviews,

[11] Jay Alammar. "A Visual Guide to Using BERT for the First Time".

[12] Alex Wang and Kyunghyun Cho. "BERT has a Mouth, and It Must Speak: BERT as a Markov Random Field Language Model".

we decided to train our fake review detection model first on the n-gram text and then on the GPT-2 text to see how the model performs when trained on fake review text of different quality. In more detail, we first combined our n-gram fake reviews with 6000 of our real reviews from our original dataset, labeled the data, and then ran them through our models. Once we tested the model on our test data, we repeated the process with the GPT-2 fake reviews.

In our model architecture, for both the n-gram and GPT-2 neural networks, we had one LSTM hidden layer along with 3 additional hidden dense layers and one final output layer. For our hidden layers, we used the swish activation function and a dropout rate of 0.7 to avoid overfitting our data, and for our output layer, we used the sigmoid activation function and the binary cross entropy loss function. Similar to our sentiment analysis component, we also played around with using different text tokenizers, using trained weights from BERT and DistilBERT, and experimenting with different loss and activation functions and number of layers. We also had a similar issue to our sentiment analysis model, though, in which the models trained on the BERT and DistilBERT weights only trained and tested on 700 reviews total each. This limitation was due to lack of processing power, RAM, and time. Aside from this limitation, for our hyperparameters, ultimately, we decided on using the sigmoid activation function and binary cross entropy loss function since we were dealing with a binary classification problem, and even with a few layers, we surprisingly reached high scores for our models.

To evaluate the accuracy of our model, we compared our model to human evaluation scores. We created 4 different versions of a fake review detection survey which we sent out on various channels of our own social network. The first three surveys contained 25 reviews each (17 real, 8 fake), and the last survey contained an additional real review (18 real, 8 fake). At the beginning of each survey, we included a brief introduction which contained a definition of what makes up a fake review—reviews generated by AI or humans to inflate or deflate an establishment's rating. We kept the introduction concise in order to reduce potential unconscious bias, so that the respondents did not begin to overthink and confuse themselves.

Results and Discussion

Sentiment Analysis

Overall, even with a simple neural network structure, we achieved highly accurate results for our sentiment analysis of the Yelp reviews as seen in Table 1 in the appendix. Our best performing model seemed to be both logistic regression and 4 layer LSTM using the trained weights from DistilBERT. Due to the randomness in the training of the models, the accuracy of our neural network models would vary slightly, but overall they performed within the range of 88 to 93 percent accuracy. Another key point is that despite the imbalance distribution of positive to negative reviews in the dataset, the models performed with high accuracy, and when we tried to add class weights to address the imbalance, the models actually performed worse. Also, given the simplicity of the task, we did not see a need to add additional layers or even

use a more complex neural network structure. Even without an LSTM, the model performed quite well in classifying the text.

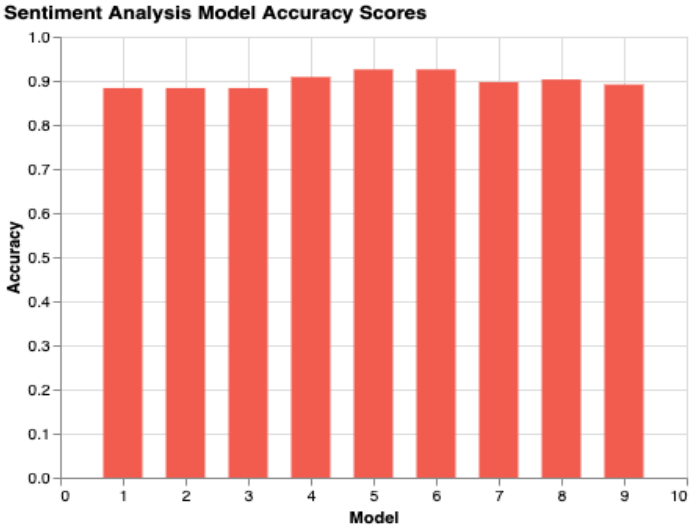

Fig. 1 Sentiment Analysis Models Accuracy Scores [13]

N-Gram Model

The n-gram models performed well across both the neural network structures and logistic regressions. These scores are relatively higher than the GPT-2 models likely because in comparison to both the human created fake reviews and the GPT-2 generated reviews, the n-gram reviews are less complex and more unrealistic. Although in Table 2 the logistic regression using BERT weights with a C value used from GridSearch has the highest accuracy, this model had the most variability when rerun with a large range in accuracy scores. Therefore we chose the 4 layer LSTM using the DistilBERT tokenizer as our best performing model to move forward with for our human evaluation score comparison.

GPT-2 Model

The training data we created (as discussed in the methods section) was run through a neural network with an LSTM layer as a baseline model. We tested variations of this neural network using DistilBERT and its weights with logistic regression, using just the DistilBERT weights, and using BERT with logistic regression, which are shown in Table 3. The model utilizing BERT with logistic regression produced the highest accuracy out of the 3 models we tested, with an accuracy of 0.8571 and a loss of 0.4384.

Using our highest accuracy model, we compared the results to the accuracy scores from our surveys in which we asked human individuals to determine whether a review was fake or not based on the text. We compared our model against each of the 4 surveys we conducted, and the results are summarized in Table 4. The GPT-2 model consistently scored higher in accuracy than human evaluators in all surveys.

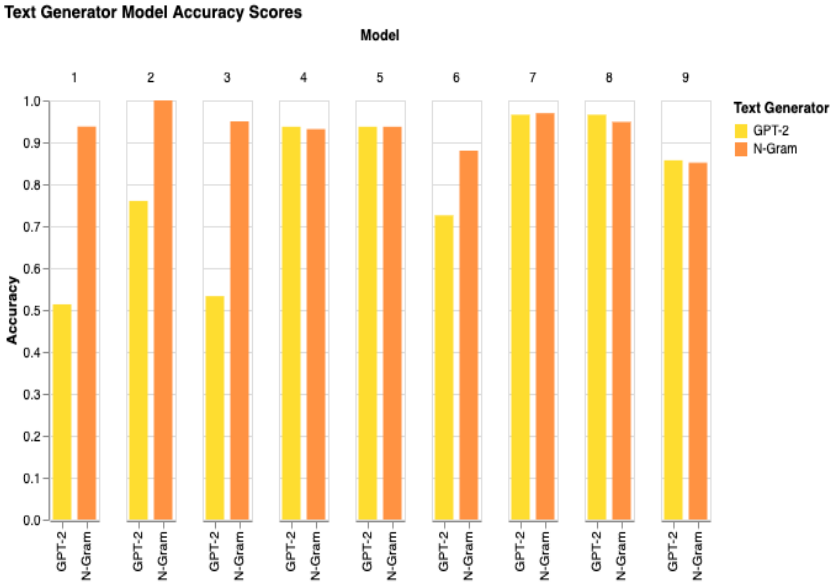

Fig. 2 N-Gram and GPT-2 Models Accuracy Scores [14]

[13] Refer to Table 1 in the appendix to see which Model corresponds to which model architecture

[14] Refer to Table 2 & 3 in the appendix to see which Model corresponds to which model architecture

Comparison with Human Evaluation Scores

Both models performed quite well when we evaluated on the same test reviews from our four surveys as seen in Table 4. Both models actually outperformed our sample of human respondents with both models reaching an accuracy between 60 and 76 percent, whereas the human evaluation scores ranged between 54 and 60 percent. Although the two models perform relatively similarly and outperform the human evaluation, the n-gram model performed better than the GPT-2 model. We speculate that this is because the n-gram model pulled directly from fake reviews we had written and the real reviews from the Yelp dataset. In contrast, the GPT-2 model pulled from both its pre-trained vocabulary as well as our fake and real reviews. Also, while the n-gram reviews had punctuation errors, because we tokenized our test reviews, this removed the errors. If these punctuation errors were left, it is possible that the n-gram model could do worse or even better, since the punctuation errors were missing in the test fake reviews we had written ourselves. Nonetheless, both models did a strong job at detecting fake reviews.

Conclusion

Our ability to produce relatively high accuracy scores for our sentiment analysis models and fake review detection models with N-Grams and GPT-2 was promising. Although not perfect, we were able to effectively emulate a variety of ways to detect fake reviews based on existing research. However, the lack of accessible labeled data with reviews that were fake or real was a challenge we had to work around. To mediate this, we attempted the task of creating fake reviews that sounded realistic enough to be believable. While we tried to use BERT as a language generating model, we were unable to successfully do so due to the fundamental nature of BERT being meant for natural language processing, *not* generation. Thus, we found it especially difficult to utilize BERT with new data. We resorted to handwriting fake reviews, which were biased in that we tend to write reviews that are too realistic and undetectable as fake by human evaluation or deep learning models. But we attempted to mediate this by bootstrapping our fake reviews with the Yelp reviews and running them through our n-gram and GPT-2 models.

Based on our research, we understand that Yelp has an internal dataset that has labeled fake reviews, so we believe that further steps can be taken using this data to generate a more accurate representation of real fake review detection and apply it to Yelp's dataset for additional testing. Additionally, some of the fake reviews generated by our models clearly were unreadable or incomprehensible, and so further fine tuning our language generation models would be another improvement. Lastly, we would hope to see a model that would detect certain features that are representative of fake reviews (i.e. strange punctuation and symbols, inconsistent capitalization, etc.) without having to preprocess data to emulate actual fake reviews. While more steps can be taken, our research has shown that NLP methods prove quite adept at detecting fake reviews and are ready for use in real-world applications.

Appendix

Table 1. Training Size and Accuracy of Sentiment Analysis Models.

Model	*Training Size*	*Loss*	*Accuracy*
4 layer simple NN, keras tokenizer	6000	0.2330	0.8835
4 layer LSTM, keras tokenizer	6000	0.2330	0.8835
4 layer LSTM, DistilBERT tokenizer	6000	0.3599	0.8835
Logistic Regression (C=0.0001, with GridSearch), DistilBERT weights	525	2.7632	0.9086
Logistic Regression (C=1), DistilBERT weights	525	2.5658	0.9257
4 layer LSTM, DistilBERT weights	525	0.2830	0.9257
Logistic Regression (C=0.0001, with GridSearch), BERT weights	525	4.1447	0.8971
Logistic Regression (C=1), BERT weights	525	3.5526	0.9029
4 layer LSTM, BERT weights	525	0.3311	0.8914

Table 2. Training Size and Accuracy of N-Gram Models.

Model	*Training Size*	*Loss*	*Accuracy*
4 layer simple NN, keras tokenizer	6000	0.1010	0.9375
5 layer LSTM, keras tokenizer	6000	1.0000	1.0000
5 layer LSTM, DistilBERT tokenizer	6000	0.0893	0.9500
Logistic Regression (C=5.2632, with GridSearch), DistilBERT weights	525	2.3684	0.9314
Logistic Regression (C=1), DistilBERT weights	525	2.1710	0.9371
5 layer LSTM, DistilBERT weights	525	0.2800	0.8800
Logistic Regression (C=15.7896, with GridSearch), BERT weights	525	0.9868	0.9695
Logistic Regression (C=1), BERT weights	525	1.1842	0.9486
5 layer LSTM, BERT weights	525	0.3412	0.8514

Table 3. Training Size and Accuracy of GPT-2 Models.

Model	*Training Size*	*Loss*	*Accuracy*
5 layer simple NN, keras tokenizer	6000	0.6927	0.5135
5 layer LSTM, keras tokenizer	6000	0.4949	0.7601
5 layer LSTM, DistilBERT tokenizer	6000	0.6847	0.5330
Logistic Regression (C=5.263252631578947, with GridSearch), DistilBERT weights	525	2.1710	0.9371
Logistic Regression (C=1), DistilBERT weights	525	2.1710	0.9371
5 layer LSTM, DistilBERT weights	525	0.5896	0.7257
Logistic Regression (C=5.263252631578947, with GridSearch), BERT weights	525	1.1841	0.9657
Logistic Regression (C=1), BERT weights	525	1.1841	0.9657
5 layer LSTM, BERT weights	525	0.4384	0.8571

Table 4. Comparison of N-Gram and GPT-2 Neural Network Models to Human Evaluation Scores.

Survey	*N-Gram Model* *(NN using DistilBERT tokenizer)*	*GPT-2 Model* *(NN using BERT-trained weights)*	*Human Evaluation*
2	0.7200	0.6800	0.5675
3	0.6800	0.6000	0.5480
4	0.6800	0.6000	0.5418
5	0.7692	0.7308	0.6044