Fake it Till You Make it: Fake Review Detection and Generation

Jose Miguel Giron
jmgiron@stanford.edu

Jonathan Gomes Selman
jgs8@stanford.edu

Nikita Demir
ndemir@stanford.edu

Motivation

- Increasingly, online reviews have become customer's primary influence in their decision making. Research has shown that a 1 star change in a business' reviews translates to anywhere between a 5 and 9 percent change in revenue. This could mean the difference between a profitable business and a failed endeavor.
- However, the nature of online ratings means that it's easy to manipulate reviews and businesses have been quick to notice. Sites have popped up online that promise gleaming reviews for a person's business or worse, negative reviews for the competition.
- There is a need for algorithms that can reliably detect when a review is fake, but the challenge lies in that most of these reviews are created by humans and therefore nearly indistinguishable.

Data

- **Yelp Zip Dataset:** 61,541 fake and not fake yelp reviews for restaurants in Chicago, where suspicious reviews are detected by Yelp's proprietary filtering algorithm.
- Review Metadata Tags:
  - Reviewer id Product id
  - Date Rating
- Data Imbalance: 13.22 % fake reviews - thus we employ data undersampling to create an evenly split dataset of 16282 reviews.
- **Yelp Dataset Challenge:** Corpus of 3.5 million restaurant yelp reviews which we used for context specific review generation.
- Create parallel dataset of context -> review, where the context for a restaurant review follows: **[rating, name, city, state, tags]**
- Tokenize reviews to remove non-ascii characters, extra whitespace, and separate out punctuation.

Related Works

1. A. Mukherjee, V. Venkataraman, B. Liu, and N. S. Glance, "What yelp fake review filter might be doing?" Proceedings of the 7th International Conference on Weblogs and Social Media, ICWSM 2013, pp. 409-418.
2. Y. Kim. "Convolutional Neural Network for Sentence Classification." arXiv:1408.5882
3. A. Joulin, E. Grave, P. Bojanowski, T. Mikolov. "Bag of Tricks for Efficient Text Classification." arXiv:1607.01759
4. M. Juuti, B. Sun, T. Mori, N. Asokan. "Stay On-Topic: Generating Context-specific Fake Restaurant Reviews." arXiv:1805.02400

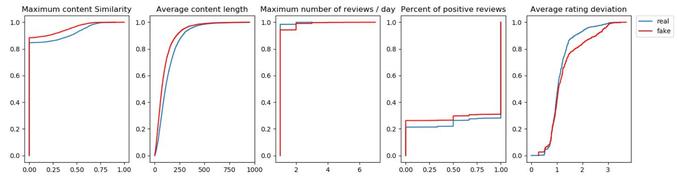
Features

Word features

- For our simpler models (SVM, Linear Regression, Naive-Bayes), we use unigram count vectors to represent each review. We saw little improvement from more complex techniques, e.g. Tf-idf vectors.
- For the more complex models (LSTM, CNN, FastText), we used pre-trained word embeddings (Glove-100d) to represent our documents and further updated the word embedding during training
- The CNN architecture used filters with height 3, 4, and 5 which roughly translates to looking at 3, 4, and 5 grams in the data and trying to determine the most significant ones.

Behavioral Features

- From the conclusions of Mukherjee et al. [1], we used 5 behavioral features but found they weren't very significant.

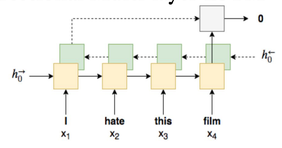


Models

Baseline Models

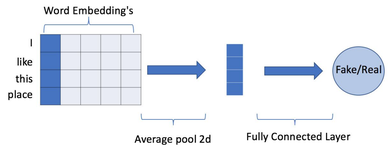
We trained 4 baseline models: Multinomial Naive Bayes, SVM with linear kernel, Random Forest, and Logistic Regression.

Bidirectional Multi-layer LSTM



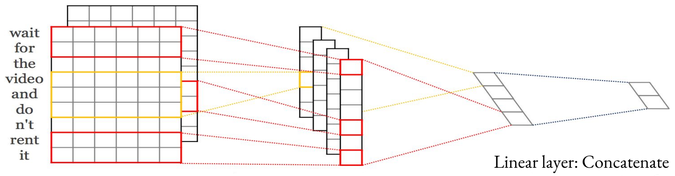
We use a 2 layer, bidirectional lstm with 512 hidden units in each lstm cell. Glove 100d vectors are used to initialize the embedding layer and a single linear output layer on the concatenated forward and backwards hidden states. We optimize BCELoss with Adam optimizer and dropout = 0.5

Fast-Text Classifier



We use word-embeddings to represent words and average across each embedding dimension to get a concise representation of the document. Then run that through logistic regression. In each step, update word embeddings and model weights with BCE loss.

CNN for Text Classification:



Conv-layer: each filter acts like an n-gram of a specific size (3, 4, 5).

Max-pool layer: picks most significant n-gram

Linear layer: Concatenate max-pool outputs into feature vector for linear layer. Use to produce prediction.

Results and Analysis

Models	Accuracy	F1 Score
Naive Bayes (word)	0.665	0.595
Naive Bayes (word + behavioral)	0.678	0.676
LR (word)	0.660	0.660
LR (word + behavioral)	0.630	0.630
RF (behavioral)	0.650	0.645
RF (word + behavioral)	0.620	0.618
FastText	0.680	0.649
CNN	0.688	0.628
LSTM	0.644	0.617

Ran using 5-fold cross validation on the Chicago Yelp dataset. Behavioral refers to the 5 behavioral features from Mukherjee et al. [1]

Discussion

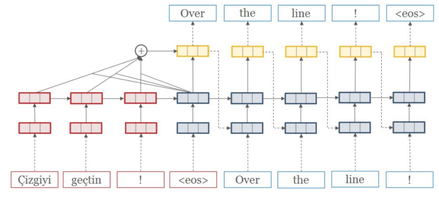
Overall our classification results were less than we were hoping, but given the difficulty of the task they were somewhat impressive. The difference between real and fake reviews is extremely difficult to discern by eye, and our human baseline only averaged around ~50% i.e. random guessing.

All of our models performed similarly even when behavioral features were introduced. This may be because there is an inherent upper limit given the problem's difficulty.

In addition, our deep models' training accuracy approached 100% which suggested we were overfitting the data, so we employed early stopping.

Generation

Given our lackluster detection results, we thought it might be interesting to attempt generating fake context-specific reviews using a neural machine translation (NMT) model. We used a large dataset of ~3 million Yelp restaurant reviews and the OpenNMT framework to generate some very fruitful results.



Examples (20 word limit)

('2', 'Bites', 'and', 'Brews', 'Pittsburgh', 'PA', 'Pizza', 'Beer', 'Bar', 'Nightlife', 'Barbeque', 'Bars', 'Chicken', 'Wings', 'Restaurants')
The pizza was good , but the service was slow and the beer selection was limited .

('5', 'Pony', 'Express', 'Pizza', 'Stow', 'OH', 'Restaurants', 'Pizza')
This is the best pizza I've had in the area . The prices are reasonable and the staff is friendly .

Future Work

- Extend the performance of our context specific review generation model by encouraging the generation of novel reviews/phrases. We will consider methods used by M. Juuti et. all [4]to introduce variability in review generation by penalizing repeated words, decreasing the log probability of generated phrases, and introducing common spelling errors.
- Further analyze metadata tags and the predictability of different reviewer based tags. We believe that metadata tags are much more informative than strictly language based models .