Fake Review Detection Using Review and Behavioral Features

Settasit Chitphentom, Nonthapat Aviphant, Anders Rikvold, Dominik Maszczyk and Nanna Barnkob

***Abstract*—In order to keep online reviews a trustworthy source of information utilized by consumers to make smarter and better decisions, we developed an effective machine learning algorithm that can classify reviews as either fake or real. We propose a model that extracts both review and reviewer information for classification. The reviews are preprocessed and vectorized by N-gram, word2vec and combined in various stemming models, and the performances using the different methods are compared in multiple experiments. For user information analysis, we introduce 3 behavioural features as well as build upon previously extracted features. The reviews are classified by a neural network model and combining the behavioral features with the content-based features finally exhibits an improvement in accuracy compared to baseline.**

I. Introduction

A significant increase in the use of online opinion sharing platforms such as Amazon, Yelp, TripAdvisor and TrustPilot has created a new way for businesses to improve their reputation and thus increase their profit. Unfortunately, some companies have seized the opportunity to boost the popularity of their products or services by paying for fake positive reviews written by opinion spammers. Furthermore, bad reviews can also be written by rivals attempting to game the system and competition. This has raised the need for quick and efficient detection of fake reviews in order to keep the Internet a trustworthy source of information.

The most simple approach to human based classification has shown to result in poor accuracy and non-uniformity that is out-competed by the machine that is able to utilize sharp and sensitive analytical tools. Thus, as an alternative solution, the problem can be stated as a matter of classifying the reviews as either fake or real using machine learning methods. The problem is approached as a supervised learning problem, needing labelled data for training and evaluating a model.

The previous attempts at solving the matter display a variety of different approaches which can generally be divided into focusing either on the reviewer or the review. This implies analysing either user behaviour or the content of the review using natural language processing (NLP), which includes preprocessing the raw text, which also can be done using various techniques. The two general approaches are based on completely different features and face distinct difficulties when attempting to fit a classification model.

Our work proposes a model combining the two previously autonomous approaches by using the direct connection between metadata and review.

The contributions of our work are as follows

1) We compare different text preprocessing techniques in order to find the one capturing the most semantic features of the review content and evaluate the differences in the performance of the model.
2) We combine feature extraction of behavioural features from metadata and textual content features of a review in a standard neural network classification model and are hereby able to create a model that exhibits higher accuracy than the baseline.

II. Related Work

A challenge when working with text-based data is how to structure it to obtain a low-dimensional model while still retaining as much of the emotional, semantic, and linguistic features of the document. To capture and use the semantic features of the reviews to predict the authenticity, Ren et al. did a content-based analysis of the reviews using word-embedding and convolutional neural network for sentence representation and a bi-directional gated recurrent neural network for combining the sentence representations to a document vector [1]. By doing this, their model exhibited a higher prediction accuracy than previous work. This is the most recent work and is a good example of the progress in text analysis capturing previously only features detectable by the human mind.

In the work by Sun et al. word embedding integrated with products data, bigram, and trigram and was used for sentence representation [2]. They recognized the fact that the pattern in the use of certain words related to the product should be acknowledged and exploited in the content analysis. For example are spammers also more likely to use sentimental words as realised by Li et al., who also were able to extract a feature of the negative/positive word ratio in fake reviews compared to 'regular' text. This was denoted the emotion model and was used with SVM [3].

Sun et al. also used two SVM classifiers in a bagging model and were finally able to achieve good accuracy. Their work will be elaborated further as we use some of this work as basis for our own, as it has also been done by the many research groups working on the same problem.

To analyse the reviewer, Fei et al. (2013) mainly used behavioural features related to the burstiness of fake review occurrences. Bursty behaviour implies that an opinion spammer is likely to write many reviews in a small time bin which is a confirmed pattern. With this in mind, the amount of reviews written in bursts and how bursty a reviwer is used as features. Hereby Fei et al. worked on the basis of metadata and incorporated the timing of the reviews using Kernel Density

Estimation and based a Markov Random Field model on this [4].

Since the previous work has contributed with stand-alone ways to solving the problem, our work proposes a way to combine and improve by using them in combination.

III. Approach

Our approach for creating an improved model is split into three parts. First, we replicated the model of Sun et al. [2], which provided us with a product to build upon. By training their model with the same data as our own, we also obtain a baseline against which to compare the performance of our model. Second, we extract behavioural features from the meta-data related to each review, by following the method described in Fei et al. [4]. Third, we integrate the extracted behavioural features into the replicated model of [2]. This should increase the performance of the model since the extracted behavioural features are largely independent of the content-based features of each review.

A. Model Replication

In their paper, Sun et al. [2] propose a novel model for detecting fake reviews based on review content-related features. Their model is based on a Convolutional Neural Network (CNN), incorporated in a bagging model with two other SVM classifiers.

To use the CNN model, the authors have to perform kind of preprocessing to prepare the data. In this case, they are using a Product Word Composition Classifier (PWCC), which assesses the polarity of each review. The output of the PWCC is then used by the CNN classifier to predict the realness of reviews.

In order to reduce the possible effects of overfitting and high variance of the CNN model, the CNN is incorporated into a bagging model. The other two classifiers in the bagging model are Support Vector Machines (SVMs), based on a BIGRAM and TRIGRAM text preprocessing, respectively. For our replicated model, we are making modifications to the original one. For the preprocessing, we are using not using a PWCC as in the paper, but are instead using several different preprocessing techniques, and comparing the difference between them. We are using a standard neural network instead of a convolutional one. Since Sun et al. does not mention any particular advantages of using a CNN over simple neural networks, we chose the simpler alternative.

Another difference between the model of [2] and ours is that we do not implement the bagging model. We justify this by hypothesising that if we can improve the classification of the neural network classifier, then the method including bagging model will also perform better. Our goal is therefore limited to improving the accuracy of the classifier itself.

B. Extraction of behavioural features

The paper of [4] describes a way of extracting a set of behavioural features from the metadata of reviews, some of which relies on an analysis of the burstiness of the data. In this paper, we perform such an analysis, and use it to extract (i) Burst Review Ratio (BBR) and (ii) Reviewer Burstiness (RB). Furthermore, we also extract the (iii) Rating Deviation (RD) of users. In addition to this, we also use 15 other behavioural features from [5]. These are as follows.

- MNR
- PR
- NR
- RD
- avgRD
- WRD
- ERD
- BST
- ACS
- MCS
- Rank
- EXT
- DEV
- ETF
- ISR

For more details on each of these features, please refer to [5].

1) Reviewer Burstiness Analysis: For analysing the burstiness of reviewers, the authors are using a Kernel Density Estimation (KDE) technique with a Gaussian kernel and peak detection to obtain bursts in reviews for each product. Each review is then marked by whether it is part of such a burst or not. The burst detection method is briefly restated here for convenience. For more detailed information, please refer to [4].

The first part of the burst detection algorithm is to obtain the reviews of each product. Then, for each product, the lifespan of the product is divided into a number of equally sized *bins* of size BSIZE. The reviews are then placed into these bins, depending on when they were created. In the paper, the authors set BSIZE to two weeks, and we do the same.

After the reviews have been placed into bins, a KDE is performed for each product, resulting in a density function for the reviews of each product. Then, a peak detection algorithm is performed on these functions to obtain the bursts. This algorithm works by taking the derivatives of the density function to identify maximums. The bins corresponding to these maximums are a part of a burst for this product. Also, the bins next to the maximums are included in the burst, as long as the number of reviews in the bin is higher than the average number of reviews in each bin for the product in question. Each review included in a bin identified as a burst is considered part of a burst for the product in question We have listed each of the extracted features below, along with a short explanation. For further information on the features, please refer to [4].

Burst Review Ratio (BBR) Measures the ratio of a reviewers reviews that appears in a burst, compared to the total number of reviews for the user. The BBR of reviewer a is calculated as $BBR(a) = \frac{|B_{a*}|}{|V_{a*}|}$, where B_{a*} is set of reviews that reviewer a wrote that appeared in review bursts, and V_{a*} represents the set of all review written by a.

Reviewer Burstiness (RB) A measure combining the bursts of products with the bursts of the reviewer itself. In principle, this feature is the same feature as *BST* of [5], and is collected from the same paper ([4]). However, we calculate the RB as a result of a burstiness analysis, which the authors of [5] seem to skip. The burstiness of reviewer a is calculated as

$$RB(a) = \begin{cases} 0 & \text{if } L(B_{a^*}) - F(B_{a^*}) > \lambda \\ 1 - \frac{L(B_{a^*}) - F(B_{a^*})}{\lambda} & otherwise \end{cases}$$

where $L(B_{a^*})$ and $F(B_{a^*})$ represents the latest and earliest time of the reviews that reviewer a wrote that appears in a burst, respectively.

Rating Deviation (RD) Measures the average deviation of a reviewers rating of a product, to the average rating of that product. The greater the rating deviation, the higher the likelihood of the reviewer writing fake reviews. It is calculated as $RD(a) = avg\frac{|r_{ap} - \bar{r}_{ap}|}{4}$, where r_{ap} refers to the rating given by reviewer a towards product p, and $p \in P_a$ where P_a is the set of products a has reviewed.

C. Integration of behavioural features

When combining the extracted behavioural features with the original model of [2], the extracted behavioural features are appended to the already existing feature vector. So if the feature vector of the neural network classifier is the PWCC output for each review, the behavioural features are simply appended to this.

IV. Experimental Setup

To perform our experiments, we needed a review dataset which consisted of not only review contents, but also metadata related to the users who wrote it. Our approach implies that there are no anonymous reviews. Luckily, the Yelp company is publishing their data for usage in research, and their dataset meets the requirements for use in our project. The dataset is provided and previously used by Rayana and Akoglu [5].

The used dataset includes 608,598 reviews from 5,044 restaurants. This amount of data should be sufficient to acquire accurate results from deep learning algorithms. As we only use reviews of restaurants and thus only have one product category, there is no need to use advanced algorithms of reducing product bias like the product word composition classifier (PWCC) utilized by Sun et al [2]. Instead, we are able to directly rely on the words frequencies in text and in addition deploy advanced stemming algorithms to further increase our accuracy.

A. Semantic feature extraction

The use of stemming algorithms is one of our improvements compared to Sun et al [2] work. To extract semantic features from the review content in the most valuable way, we perform our experiments with 4 different ways of preprocessing.

1. **Basic preprocessing (baseline)** - This was the base for all following preprocessors. First of all, we unify all signs to simple ASCII codes to avoid any problems with encoding. A single word is matched as at least two letters next to each other. Then the top 5% most popular words are filtered out as a simple probabilistic solution for getting rid of most stop words. Also, words occurring less than 10 times are filtered out to reduce ”long tail” in our features distribution.
2. **Preprocessing + Porter stemming** - This follows the same steps as in basic preprocessing, with the difference that words can now contain apostrophe.
   We then follow the Porter stemming algorithm for each word. This means that the word endings are stripped based on grammatical rules, for example when shortening ”nicely” to ”nice”. As a result of this, different forms of words of the same meaning become the same features. [6]
3. **Preprocessing + Lancaster stemming** - This uses the same preprocessing as Porter, but with a Lancaster stemmer (also called Paice/Husk algorithm).
   Porter algorithm being a precursor in the field after many years is doing reasonably well but over the time new algorithms arose like Lancaster stemming. This algorithm is more ”aggresive” then Porter stemming. It is more optimised to cut rated to much from the word ending rather than omit the word. [7] It cannot be said that it in general performs better than Porter, since the performance of the two algorithms is extremely load dependent. There is no deterministic approach to describe their performance based on load instead of choosing the better one is usually made by testing. [8]
4. **Preprocessing + Porter2 stemming** - This uses the same preprocessing as in number 2) once more this time with an improved version of Porter algorithm, sometimes called Porter2.
   It was fully rewritten by using ”Snowball” string processing language and enriched by many small improvements which author found important after 20 years of usage of its basic form. [9]

Apart from text preprocessing for N-gram models, we utilised word embedding from Word2vec model [10]. We used only 100 dimensions from 300 dimensions to compare with N-gram model. Document representation is simply the average of the words in the sentences.

Used parameters for clearance has been shown in table I.

	Basic	Porter	Lancaster	Porter2
ASCII unification	✓	✓	✓	✓
filter out top 5%	✓	✓	✓	✓
filter out stop words (dictionary)				✓
filter out less than 10 occurences	✓	✓	✓	✓
word match consist of two and more letters	✓	✓	✓	✓
word match word can contain comma		✓	✓	✓

TABLE I. Different preprocessing aproaches and used parameters

B. Behavioural feature extraction

Behavioural features greatly described in section III-B are all extracted from meta-data of each following review. Which results in 18 features which only partly overlaps and are characterised by great correlation to the fakeness of reviews.

C. Ensembling & classifying

100 the most popular features from semantic extraction and 18 behavioural ones are joining into a feature vector for each review. Then to be able to use standard machine learning models effectively we reduce our skew data to consist the same amount of fake and real reviews. Based on data prepared in this way we perform 5-fold k-fold feeding to Neural Network with two hidden layers, 100 nodes each with ReLu activation function, the learning rate of 0.001 with 300 maximum iterations and obtaining following results.

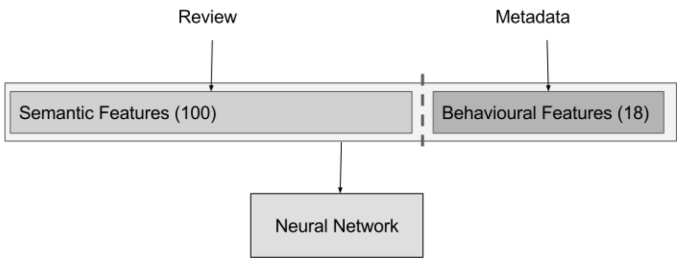

Fig. 1. Visualisation of data set

V. Results and Discussion

With the neural network model, the behavioural features without semantic features can achieve 0.67-0.68% precision on the same data-set described in Table II. The Table III shows the precision from training and testing 5-fold cross validation on YelpZip data-set. The set contains 80466 labelled as 'real' and 80466 as 'fake'. The result shows that with inclusion of behavioural data, the model can achieve roughly 10% increase of precision across all preprocessing model.

Features from [5]	Our Features
0.67	0.68

TABLE II. The precision on classifying reviews using behavioural features

Processing Model	Semantic Features	With Behavioural Features
Unigram	0.59	0.70
Porter	0.61	0.73
Lancaster	0.62	0.73
Porter2	0.65	0.73
Word2Vec	0.65	0.75

TABLE III. Precision on classifying review using various prepossessing models

We conduct the experiment utilizing multilayer perceptron with two hidden layers for classification because it can solve arbitrary problem when two-layer neural network can make only convex boundary decision. The network is optimised by gradient descent method with small learning rate. We do not apply any convolutional layer since the network get input as document representation.

We experiment on text preprocessing part by using N-gram model from [2] as a baseline. Five text preprocessing methods, including Porter Stemming, Porter2 Stemming, Lancaster Stemming, and word2vec, are used. The results show significant improvement from the baseline model. Word2vec have the best performance with 6% accuracy higher than the baseline. For behavioural features, we add three more features and achieve slight enhancement from Table II.

The data used in the experiment from Yelp is based on its classification algorithm. This, however, doesn't 100% accurately classify if the review is a scam. This is one of limitation to construct a good classification model. Another way is employ people to judge and/or write scam reviews and includes data with other legitimate model. Nevertheless, this method requires more time and resources to generate data.

VI. Conclusions

In summary, we aimed to improve the framework of the classification task. We investigated fake review detection in two aspects: behavioural features from review metadata and textual features from the review content. For semantic features of review, we experiment with different text preprocessing and achieve the best result using word2vec. For behavioural feature, we introduce 3 new features to increase the performance of the model. Importantly, we proved that by combining the two types of features we can enhance the precision of classification by between 0.07 and 0.12.

We see several avenues for future work. One way to further improve the accuracy is to improve feature extraction in the content based analysis, but also more behavioural features extracted from metadata can develop the model, but it is difficult to come up with related meaningful features. Further textual extraction may include syntactic features (e.g. part of speech tagging) and name entity. Additionally, there are various ways to create document representation vectors; for example latent semantic analysis (LSA) and convolutional neural network (CNN). For classifiers, recursive model, like recurrent neural network (RNN) and long short term memory networks (LSTM), can be utilised to explore temporal features. Moreover, new data set is always on demand in this field since there is still no existing big data with ground truth.

References

[1] Y. Ren and D. Ji, "Neural networks for deceptive opinion spam detection: An empirical study," *Information Sciences*, vol. 385386, pp. 213 – 224, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025517300166

[2] C. Sun, Q. Du, and G. Tian, "Exploiting product related review features for fake review detection," *Mathematical Problems in Engineering*, 2016.

[3] Y. Li, X. Feng, and S. Zhang, "Detecting fake reviews utilizing semantic and emotion model," pp. 317–320, July 2016.

[4] G. Fei, A. Mukherjee, B. Liu, M. Hsu, M. Castellanos, and R. Ghosh, "Exploiting burstiness in reviews for review spammer detection," *Proceedings of the Seventh International AAAI Conference on Weblogs and Social Media*, 2013.

[5] S. Rayana and L. Akoglu, "Collective opinion spam detection: Bridging review networks and metadata," *Conference on Knowledge Discovery and Data Mining (KDD)*, 2015.

[6] C. Van Rijsbergen, S. Robertson, and M. Porter, *New Models in Probabilistic Information Retrieval*, ser. British Library research & development report. Computer Laboratory, University of Cambridge, 1980. [Online]. Available: https://books.google.co.kr/books?id=WDZ3bwAACAAJ

[7] C. D. Paice, “Another stemmer,” *SIGIR Forum*, vol. 24, no. 3, pp. 56–61, Nov. 1990. [Online]. Available: http://doi.acm.org/10.1145/101306.101310

[8] A. G. Jivani *et al.*, “A comparative study of stemming algorithms,” *Int. J. Comp. Tech. Appl*, vol. 2, no. 6, pp. 1930–1938, 2011.

[9] P. M.F., “Developing the english stemmer,” 2002, accessed: 2017-06-15. [Online]. Available: http://snowball.tartarus.org/algorithms/english/stemmer.html

[10] G. C. J. D. Tomas Mikolov, Kai Chen, “Efficient estimation of word representations in vector space,” *CoRR*, 2013.