CSC 798: Research in Computer Science, Spring 2021

Instructor: Prof. Dr. Jamil M. Saquer

<u>Final Report</u>

"Classification of genuine and fake reviews using Topic Modelling, Word Embedding and Convolutional Neural Network (CNN) "

Ayman Mahmud Haque

Computer Science Department

Missouri State University

Classification of genuine and fake reviews using Topic Modelling, Word Embedding and Convolutional Neural Network(CNN)

Introduction

We have used word embeddings to train a convolutional neural network to classify between real and fake reviews from our Fake reviews' dataset. We have used the 'Ott' dataset [1] to train and test our model. We have first extracted the sentences of the dataset containing only the sentences containing our topic words. Then, we have used the 'Gensim Word2vec' library to train our word embeddings after preprocessing the dataset and used the word embeddings to transform our dataset corpora by taking the words and replacing them with their vector representations. We then take this transformed vector represented sentences as input to our convolutional neural network. We have used k-fold method to train and test our model and consequently we evaluate our model for changes in accuracy for different preprocessing techniques and changes in the network architecture.

Description of Dataset (Ott et al., 2011)

Three procedures have been used to evaluate the performance of the dataset:

1. **Standard text categorization with n-grams based classifiers.** (Joachims, 1998; Sebastiani, 2002);

2. **Psycholinguistic deception detection**: in which we expect deceptive statements to exemplify psychological effects of lying, such as increased negative emotion and psychological distancing (Hancock et al., 2008; Newman etal., 2003);

3. **Genre Identification**: where deceptive and truthful writing is classified as imaginative and informative writing, respectively.

4. **N-gram based classifiers**: SVM and Naïve-Bayes

According to Ott et al., 2011 [1], Procedures 2 and 3 where outperformed at statistically significant levels by n-gram based classifiers alone. A combined classifier of n-grams and Psychological deception features achieves almost 90% of cross-validated accuracy. Human judges perform poorly on deceptive opinion spam detection. (Bond and DePaulo, 2006)

Collection of Truthful Reviews

The authors (Ott et al.,2011) collected truthful opinions from TripAdvisor of the 20 most popular hotels in Chicago to consist the truthful opinion section of the dataset.

There are a few folds to claiming that the Truthful reviews gathered to construct the dataset (Ott et al., 2011) is mostly genuine:

1. <u>**Reviews on products with high sales ranks generally have low level of spam.**</u>

   In this paper[2], (Jindal et al.,2008) observes:

   • Products with sales rank <= 50 (number of reviews = 3009)
   • Products with sales rank >= 100K (number of reviews = 3751)

   This can be observed in the following lift curve (Jindal et al., 2008):

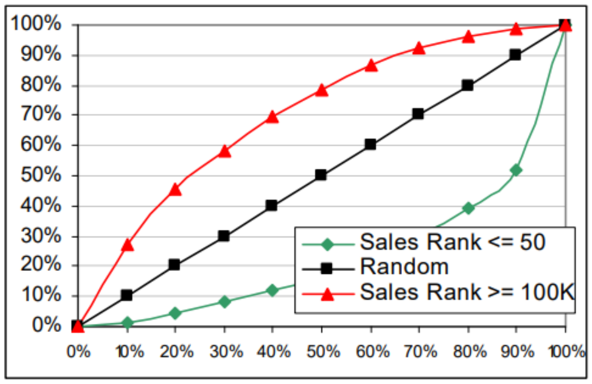


Figure 11: Lift curves for reviews corresponding to products with different sales ranks.

   To quote the author (Jindal et al., 2008):

   "We can see that reviews on products with high sales ranks generally have very low level of spam. Its lift curve is even below the random line. This is good news since it shows that spam activities are more limited to low selling products. This is quite intuitive since it is difficult to damage reputation of a high selling or popular product by writing a spam review."

   Considering these observations, the selection reviews of 20 of the top-rated hotels of Chicago from TripAdvisor seems prudent as it may contain significantly less spam reviews than from regular hotels. Yet, that cannot be the only filtering requirement for constructing a close to pristine dataset, hence (Ott et al.,2011) moves on to refining with the next steps.

2. **<u>Excluding disruptive reviews that may void the authenticity of the dataset.</u>**

   For truthful opinions, the author mines 6,977 reviews from the 20 most popular Chicago hotels on TripAdvisor. From these the author eliminates:

   • 3,130 non-5-star reviews.

   • 41 non-English reviews

   • 75 reviews with fewer than 150 characters since, by construction, deceptive opinions are at least 150 characters long.

   • 1,607 reviews written by first-time authors (new users) as these opinions are more likely to be spam, reducing the integrity of our truthful review data (Wu et al., 2010).

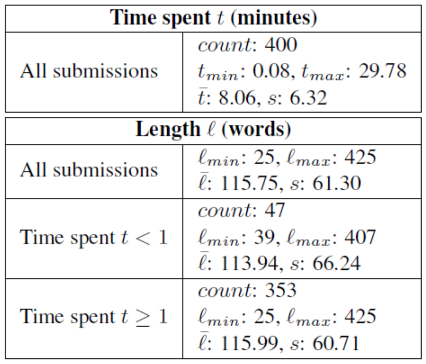
   The author balances the number of truthful and deceptive opinions by selecting 400 of the remaining 2,124 truthful reviews so that the lengths of documents of the truthful reviews are similarly distributed to the deceptive reviews. The author mentions a Work by Serrano et al. (2009) that suggests that a log-normal distribution is appropriate for modeling document lengths. Hence, the author selects 20 truthful reviews from a log-normal (left truncated at 150 characters) distribution fit to the lengths of the deceptive reviews from each of the 20 hotels. Combined with the 400 deceptive reviews yields the dataset of 800 reviews.

Collection of Deceptive Reviews

A pool of 400 Intelligent tasks (HITs) was created in Amazon Mechanical Turk and allocated across the 20 hotels. Only one submission was allowed per Turker(Online Worker) and the task was restricted to workers only in United states and strictly with approval rating of 90%. Each turker was strictly allowed 30 minutes to task. Each task presented the hotel name and website of a hotel and asked to write a fake review to be posted on a travel review website. There were strict guidelines for the concoction of reviews-

1. Must sound realistic and portray the hotel in positive light.
2. Must be for the hotel provided in the task.
3. Must be intelligible.
4. Of reasonable length.
5. Not plagiarized.

The 400 deceptive opinions have been carefully vetted and took 14 days to collect. 12% of the submissions were completed in under one minute. To verify the authenticity of fake reviews an independent two-tailed test between the mean length of these submissions with the others revealed no significant difference. (p =0.83). This led to the hypothesis that these submitters started working prior to formally accepting the HITs to circumvent the time limit.



Time spent t (minutes)	
All submissions	$count$: 400 t_{min}: 0.08, t_{max}: 29.78 $\bar{t}$: 8.06, s: 6.32
Length ℓ (words)	
All submissions	ℓ_{min}: 25, ℓ_{max}: 425 $\bar{\ell}$: 115.75, s: 61.30
Time spent $t < 1$	$count$: 47 ℓ_{min}: 39, ℓ_{max}: 407 $\bar{\ell}$: 113.94, s: 66.24
Time spent $t \geq 1$	$count$: 353 ℓ_{min}: 25, ℓ_{max}: 425 $\bar{\ell}$: 115.99, s: 60.71

Table 1: Descriptive statistics for 400 deceptive opinion spam submissions gathered using AMT. s corresponds to the sample standard deviation.

Discussion:

1. For the deceptive reviews gathered using AMT, each review had been checked for plagiarism. The same thing could have provided better authenticity for the truthful reviews.

2. The author mentions that the 5-star reviews of the 20 most popular hotels has been scraped to form the truthful reviews dataset. In the trip-advisor website, the star rating is in reference to the hotel and is an indicator to individual customer satisfaction. We have been unable to find a rating system for each review. Logically, this implies the truthful reviews dataset is skewed but that is in contradiction with our dataset which contains truthful+negative polarity reviews. Excerpt for reference below:

   "Following the work of Yoo and Gretzel (2009), we compare truthful and deceptive positive reviews for hotels found on TripAdvisor. Specifically, we mine all 5-star truthful reviews from the 20 most popular hotels on TripAdvisor[8] in the Chicago area.[9] Deceptive opinions are gathered for those same 20 hotels using Amazon Mechanical Turk[10] (AMT)."

3. Author mentions Yoo and Gretzel (2009) while describing the dataset for truthful reviews but does not explicitly mention the extent of the work that was adapted from that in creating and labeling the dataset (please refer to the excerpt at point number 2 from above)

Preprocessing the Data

Before training our word-embeddings and training our model, we perform various preprocessing methods on the dataset for acquiring better accuracy. First, we used the Topic modelling using LDA to generate topic models from the corpus. Using these Topic words, we extract only the sentences containing the topic words to form our dataset. Using the 'NLTK' library to remove stop words and punctuation from the dataset. We use this preprocessed dataset to train the word2vec model to find word embeddings.

Word Embedding Using Word2Vec (Skipgram)

Word Embedding is an important technique in natural language processing. It is a representation of words where the words are mapped to vectors of real numbers and similar words are clustered together mapped to similar vectors in a predefined vector space. Word embedding captures the semantic and syntactic similarity in relation to other words in a corpus.

We have used Word2vec which is an unsupervised learning method which learns word embeddings for texts. It has two different learning models namely, C-BOW and Continuous Skipgram model. We have used the Skipgram model which is a two layered neural network unsupervised learning model to train our word embeddings. The whole dataset is used to train the word embeddings. The dataset is then converted from words to their corresponding word embedded vectors to train the Convolutional Neural Network and evaluate the model.

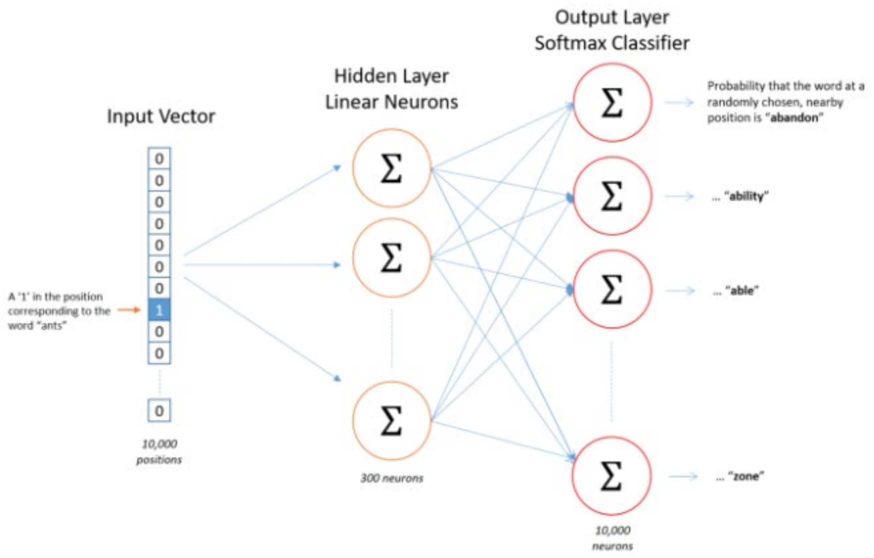
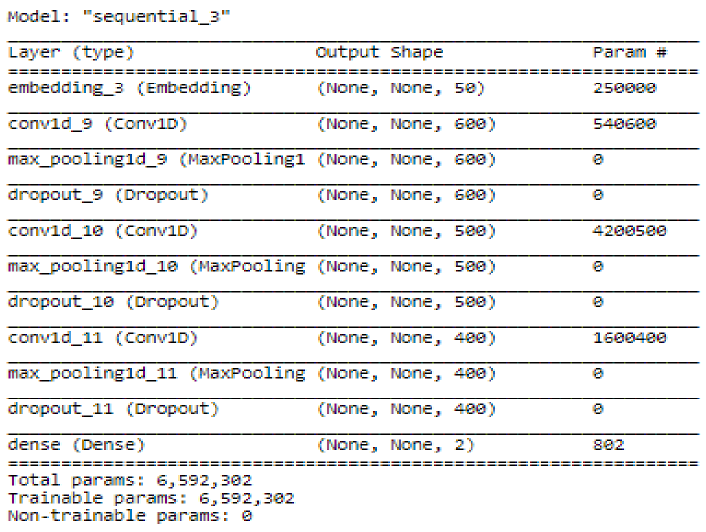


FIGURE 1 SKIPGRAM ARCHITECTURE (SOURCE: BECOMINGHUMAN.AI)

Convolutional Neural Network (CNN)

We use TensorFlow and Keras sequential model to build our CNN architecture. The model consists of an embedding layer where we forward our trained vector space of word embeddings. The embedding layer is connected to three CNN layers 600,500 and 400 layers respectively. The filter sizes are 18, 14 and 8. The activation function at each layer is reLU. Each of the CNN layer is followed by a max-pooling layer and a dropout layer. The pool sizes for the first two Max-pooling layer is 6 and the later is 1. The dropout



```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_3 (Embedding)      (None, None, 50)          250000
_________________________________________________________________
conv1d_9 (Conv1D)            (None, None, 600)         540600
_________________________________________________________________
max_pooling1d_9 (MaxPooling1 (None, None, 600)         0
_________________________________________________________________
dropout_9 (Dropout)          (None, None, 600)         0
_________________________________________________________________
conv1d_10 (Conv1D)           (None, None, 500)         4200500
_________________________________________________________________
max_pooling1d_10 (MaxPooling (None, None, 500)         0
_________________________________________________________________
dropout_10 (Dropout)         (None, None, 500)         0
_________________________________________________________________
conv1d_11 (Conv1D)           (None, None, 400)         1600400
_________________________________________________________________
max_pooling1d_11 (MaxPooling (None, None, 400)         0
_________________________________________________________________
dropout_11 (Dropout)         (None, None, 400)         0
_________________________________________________________________
dense (Dense)                (None, None, 2)           802
=================================================================
Total params: 6,592,302
Trainable params: 6,592,302
Non-trainable params: 0
```

FIGURE 2: CNN MODEL SUMMARY

layer and the early callback is introduced to prevent overfitting. The callback function monitors the sparse categorical accuracy of the model with a patience of 20, which means that the model will keep training until the accuracy has unsubstantial increase for at least 20 iterations. A dense layer with 2 neuron and a SoftMax activation serves as the output layer. We use k-fold-validation to train and test our model. We use 4 folds of the dataset to train the model and the fifth fold to evaluate. We have achieved highest accuracy of 89.33% percent with 30 percent of test data at each fold.

Evaluations

We have attained the highest Accuracy of 89.327% with the LDA+CNN+Word2vec model on a train/test ratio of 90:10 with a 5-fold cross validation. We have observed that the highest accuracy can be attained on a three-layer convolutional network described above. Increasing the filter sizes from base case did not appear to increase the accuracy, rather, increasing the filters to more than 900 on the layers decreases the over accuracy by a few percent. Introducing a callback function monitoring the validation accuracy and adding drop-out prevented overfitting. We have also observed the performance for different preprocessing techniques (figure) and changing Training and Test splits (Table:1). We have measured the performance of our CNN model against various traditional machine learning methods like Naïve-Bayes, Multilayer Perceptron and Gradient-Boosting classifier (Table:3)

Table 1: Change in accuracy for change in training and test split

Train-Test Split	**Method**	**Accuracy**
90:10	CNN + word2vec	89.327%
80:20	CNN + word2vec	87.68%
60:40	CNN + word2vec	88.04%

Table 2: Change in performance due to various Preprocessing techniques

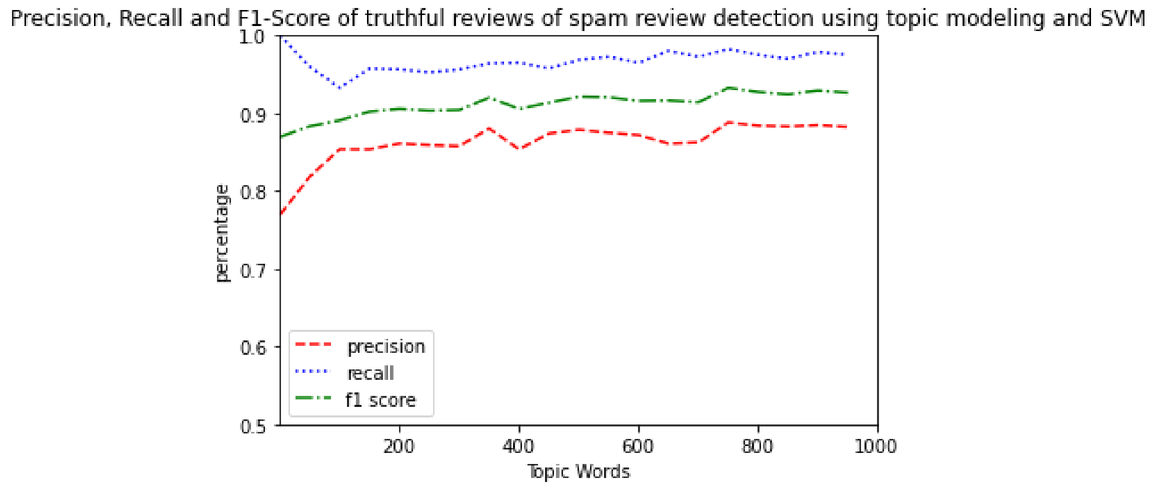
Technique	**Method**	**Accuracy**
Without stop word removal and punctuation removal	LDA + TF-IDF+CNN	64%
Without stop word removal and punctuation removal	LDA+ Word2vec+ CNN	72%
With Stop word removal	LDA + word2vec + CNN	84.186%
With Stop words and punctuation removal	LDA + word2vec +CNN	89.327%

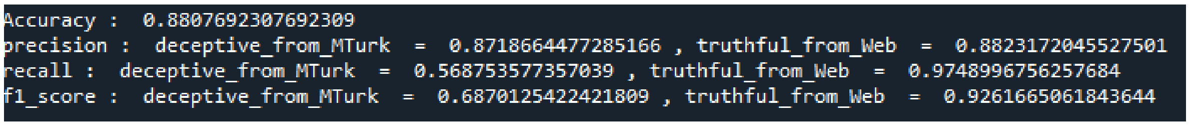
TABLE 3: COMPARISON OF PERFORMANCE AMONG DIFFERENT ML METHODS

Methods (5-fold cross-validation)	**Technique**	**Highest Accuracy Achieved**
Naïve-Bayes	Topic modelling using LDA	84%
SVM with SGD	Topic modelling using LDA	88.18%
Gradient Boosting Classifier	Topic modelling using LDA	78.812%
Multilayer Perceptron	Topic modelling using LDA	82.21%
Convolutional Neural Net	Topic modelling with LDA+ TF-IDF	62%
Convolutional Neural Net	Topic modelling using LDA + Word2vec(Skipgram)	89.325%

Precision, recall and F1 score of Traditional machine learning methods.

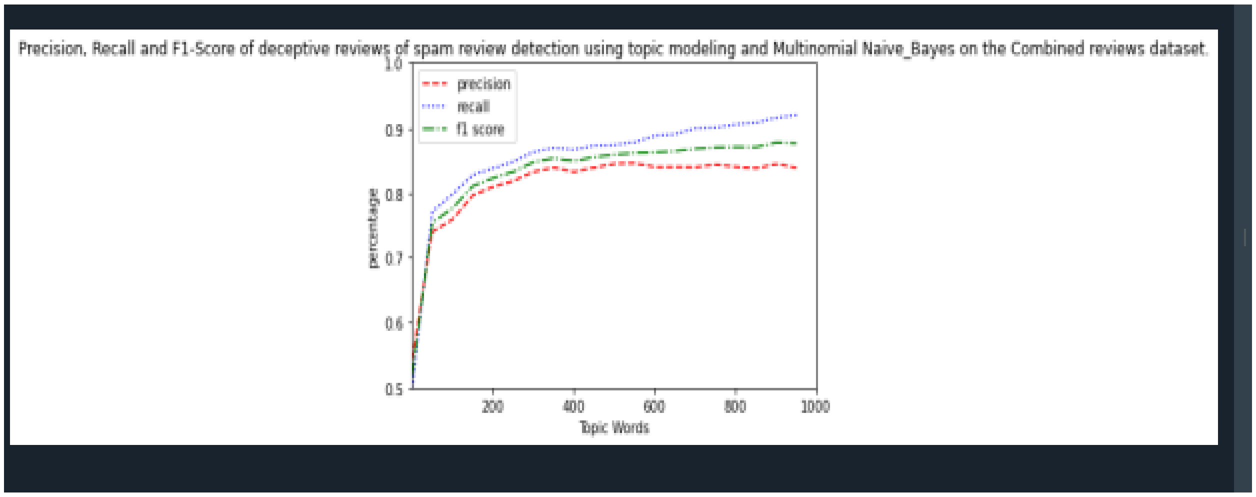
Support Vector Machine with SGD



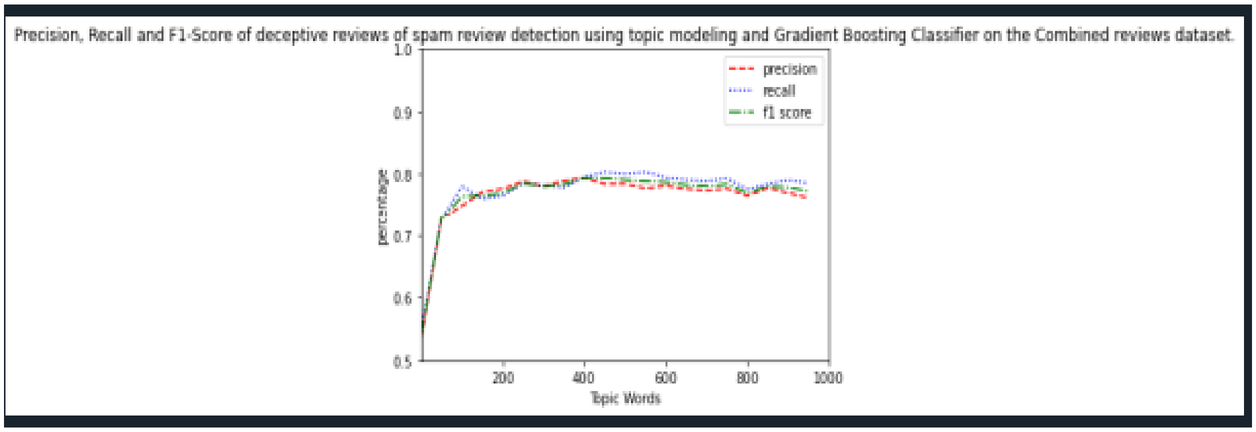


```
Accuracy :  0.8807692307692309
precision :  deceptive_from_MTurk  =  0.8718664477285166 , truthful_from_Web  =  0.8823172045527501
recall :  deceptive_from_MTurk  =  0.568753577357039 , truthful_from_Web  =  0.9748996756257684
f1_score :  deceptive_from_MTurk  =  0.6870125422421809 , truthful_from_Web  =  0.9261665061843644
```

Multinomial Naïve-Bayes

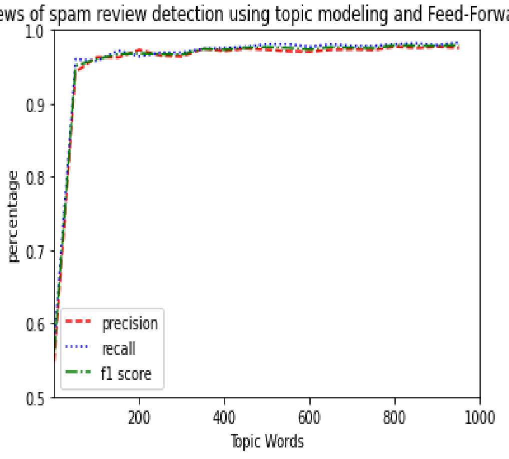

Precision, Recall and F1-Score of deceptive reviews of spam review detection using topic modeling and Multinomial Naive_Bayes on the Combined reviews dataset.

Gradient Boosting Classifier


Precision, Recall and F1-Score of deceptive reviews of spam review detection using topic modeling and Gradient Boosting Classifier on the Combined reviews dataset.

Multilayered Neural network

Precision, Recall and F1-Score of truthful reviews of spam review detection using topic modeling and Feed-Forward neural net on the Combined reviews dataset.



REFERENCES

1. M. Ott, Y. Choi, C. Cardie, J. T. Hancock, Finding deceptive opinion spam by any stretch of the imagination, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, Association for Computational Linguistics, 2011, pp. 309{319.
2. N. Jindal, B. Liu, Opinion spam and analysis, in: Proceedings of the 2008 international conference on web search and data mining, ACM, 2008, pp. 219{230.