Rupal Bhargava*, Anushka Baoni and Yashvardhan Sharma

Composite Sequential Modeling for Identifying Fake Reviews

<https://doi.org/10.1515/jisys-2017-0501>

Received August 25, 2017; previously published online April 28, 2018.

Abstract: This paper presents a comprehensive analysis and comparison of various proposed sequential models based on different deep networks such as the convolutional neural network, long short-term memory, and recurrent neural network. The different sequential models are analyzed based on the number of layers, the number of output dimensions, order, and the combination of different deep network architectures. The proposed approach is compared to a baseline model based on traditional machine learning techniques.

Keywords: Spam detection, deep learning, machine learning, fake reviews.

1 Introduction

Opinions influence almost all decisions of humans. The day-to-day choices that we make in our lives are based on how our peers and friends perceive the world. Be it an organization or an individual, opinions are an integral part of the decision-making process. The study of emotions, opinions, choices, sentiments, and behavioral patterns is called opinion mining, also known as sentiment analysis. With the growth of social media, organizations and individuals depend highly on the media content for decision making. These online reviews help customers, companies, and vendors to make decisions regarding the quality of products and services. Social media allows an individual to post any feedback, reviews, or comments regarding a product, service, or organization anonymously. Because of this anonymity, people with hidden motives or malicious intentions post deceptive opinions or feedback to misguide or wrongly influence people on social media. Such activities are called opinion spamming [16]. Such content on social media falls into the category of opinion spam, and the people are called opinion spammers. It is, thus, important to address the problem of review spam on social media. The need of the hour is to have a technique capable of analyzing the truthfulness of reviews to assist with the decision-making process or for marketing intelligence.

Product reviews are a progressive way of generating online consumer content as they offer valuable insight into user preferences and requirements. The World Wide Web contains a vast amount of user reviews and opinions on various products expressed in newsgroups, sites, etc. As a result, opinion mining has received increasing attention over the last few decades [16]. The e-commerce websites have made it a practice to let consumers voice their opinions and publish reviews for various products on their websites and/or mobile apps. These reviews give customers the analytical information to broaden their scope with more competitors who are providing the same product/services [2]. Product manufacturers get to know consumer preferences/interests, as well as the positive/negative attributes of their products and that of the competitors and, hence, can take necessary action by which profits could be maximized [7].

Several researchers have studied the problem of deceptive spam reviews [11, 21]. Various review-filtering systems have been developed and are being used by companies such as Yelp and Dianping to detect low-quality and fake reviews on their product pages. Many spammers work collectively to promote or demote a

*Corresponding author: Rupal Bhargava, WiSoc Lab, Department of Computer Science, Birla Institute of Technology and Science, Pilani-333031, India, e-mail: rupal.bhargava@pilani.bits-pilani.ac.in

Anushka Baoni and Yashvardhan Sharma: WiSoc Lab, Department of Computer Science, Birla Institute of Technology and Science, Pilani-333031, India

set of target products so that they can work without leaving a trail [22]. In the experiments of Ott et al. [29], the average accuracy of detecting spam by three human judges is only 57.33%, and hence, the research for the detection of deceptive opinion spam is meaningful and necessary. The problem of spam is troublesome and poses a security threat as well. However, it is very difficult to filter out a spam review or capture spammer behavior, even if done manually.

The objective of this paper is to distinguish whether a review is a spam or truth (not spam). The task can be transformed into a binary class classification problem. In traditional machine learning techniques, feature engineering is important. However, from a semantic perspective, the inherent law of data can hardly be learned from it. Deep learning refers to a set of algorithms that attempt to learn in multiple levels, corresponding to different levels of abstraction. It is nothing but a very large or deep neural network, which automatically learns feature hierarchies with features from higher levels of the hierarchy formed by the composition of lower-level features. Deep learning algorithms have performed well on various natural language-processing (NLP) tasks such as paraphrase detection [14], POS tagging [33], language modeling [39], and sentiment analysis [37]. This article presents an experimental study and its analysis on the variants of sequential models based on the deep network architecture. It also compares the approach and variants to that of the baseline approach based on the traditional machine learning techniques to analyze the effectiveness of the proposed approach.

2 Related Work

Social media has seen increased spamming with the advancement in technology. Many networks have tried to keep stringent policies for the safety of their customers, but spammers adapt suitably as well. In 2008, Twitter officially announced that performance of the entire system was threatened because of the Follow spam accounts (https://blog.twitter.com/official/en_us/a/2008/making-progress-on-spam.html). Chakraborty et al. [3] proposed nine domain-based classifications of spam techniques:

- (i) E-mail [36]
- (ii) Blog [17]
- (iii) Microblog [12, 13]
- (iv) Bookmarking [9, 32]
- (v) Social network [1]
- (vi) Review [10, 25]
- (vii) Location search [6]
- (viii) Comment based [34]
- (ix) Cross-media [26]

Another categorization proposed was based on Twitter by Jeong et al. [15] on twitter spam filtering, link spam filtering, and data-mining approach for spam detection [9, 18].

Existing proposed works also employed supervised learning techniques using features based on review text, rating, and other metadata [16, 29]. The social honeypot is used by Lee et al. [20] to allure spammers for building benchmark data sets. Diale et al. [8] optimized the kernel type and kernel parameters in improving the performance of support vector machine (SVM). Authors also varied the number of features for SVM, adaptive boosting, and random forest to check the effect of their performance. Mi et al. [27] applied stacked autoencoder for identifying spam detection.

Kim et al. [19] proposed two ways of spam detection: by comparing the similarity between user comments and publisher posts and by learning the single-representative meta feature such as the username or ID. Shao et al. [35] proposed a hybrid method based on image and text spam recognition. In the case of an image, local and global image features are used, whereas, in the case of text, semantic properties are used. Li et al. [24] proposed a neural network-based model to learn the representation of reviews. Authors compute sentence importance and incorporate them to represent document representation. Song et al. [38] proposed

an incremental learning approach for tackling the changing feature space and exploits a probabilistic generative model for mining latent semantics from user-generated comments. Li et al. [23] discovered that a reviewer's posting rates are bi-modal. Multiple spammers indulge in co-bursting (collectively and actively posting reviews for the same set of products in a short time frame).

3 Proposed Approach

Proposed approach comprises multiple phases as shown in Figure 1. In the pre-processing phase, the raw reviews are taken and converted into lower case. The vocabulary of the entire data set is extracted from the reviews, and the overall frequency of each word is calculated. A sorted dictionary of all the words along with their frequencies is generated, and the most frequent word is given a label 1, the second most frequent word is labeled 2, and so on. The words in the raw reviews are replaced by generated labels correspondingly, and in this way, each review is encoded as a sequence of word indexes (integers).

In the sequential model phase, different sequential models are generated for classification of the reviews using architectures explained in Sections 3.1, 3.2, and 3.3. The first layer of each model is an embedding layer, which converts positive integers (indexes) into dense vectors of fixed size. Word embeddings help in associating a numeric vector to every word in a dictionary. Ideally, words with a similar semantic meaning will be closer in the embedding space. The embedding layer [5] converts each word encoding (index) to a vector in an embedding space. The vectors are initialized randomly and then optimized in an iterative manner. The input parameters to this layer are input_dim, output_dim, and dropout. The input dimension is the vocabulary size (20,000 for all the models implemented), and the output dimension is the dimension of the dense embedding vectors (512 for all the models). Dropout is varied between 0 and 1 (0.2 for this specific model), which corresponds to the fraction of the embeddings to drop.

3.1 Recurrent Neural Network (RNN)

The recurrent neural network is a special type of neural network designed for sequence problems [39]. The RNNs are called recurrent because they repeat the same task for every element of a sequence, and the output depends on previous computations. The RNNs have “memory”, which keeps track of the information that has been calculated so far. The presence of a loop ensures information persistence and connects previous information to the present task being targeted. Figure 2 (<http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/>) shows the unrolling of the RNN. x_t is the input at time step t .

Figure 1: Block Diagram for the Proposed Approach.

Figure 2: RNN Architecture.

s_t (“memory” of the network) is the hidden state at time step t . s_t is calculated based on the input at a current step and the previously hidden state using Equation (1). The function f is a non-linear function such as $\tanh$ or ReLU . s_{t-1} is required to calculate the first hidden state and is typically initialized to all zeroes. o_t is the output at step t , which is calculated solely based on the memory at the time.

$$s_t = f(U \cdot x_t + W \cdot s_t) \quad (1)$$

An important point to note is that unlike the traditional deep neural networks, which use different parameters at each layer of the network, the RNN has the same parameters across all layers (e.g. U , V , and W in Figure 2). This means that at each step, the same task is being performed. The only varying factor is the input. This greatly reduces the total number of parameters the network has to learn.

The recurrent connections add state or memory to the network and allow it to learn broader abstractions from the input sequences. However, if more context is required, then, the RNNs are not very efficient. A fully connected RNN is used for this sequential model, where the output is fed back to an input. The parameters set for the RNN layer of the proposed model are dimensionality of the output layer (256 for the implemented model), $\tanh$ as the activation function, and the dropout (fraction of units to drop for the linear transformation of the inputs) as well as the recurrent_dropout (fraction of units to drop for the linear transformation of the recurrent state) set at 0.2.

3.2 Long Short-Term Memory (LSTM)

Long short-term memory (<http://colah.github.io/posts/2015-08-Understanding-LSTMs/>) is a special kind of RNN capable of learning long-term dependencies. All the RNNs have a chain of repeating modules. In the standard RNNs, the repeating module has a very simple structure, such as a single $\tanh$ layer. In the repeating module of the LSTMs, there are four layers interacting in a very special way.

The LSTM has a cell state, to which information can be added or removed with the help of gates. Each gate has a sigmoid neural net layer that outputs a value between 0 and 1 (0 means let nothing through, whereas 1 means allow everything to pass through) and a point-wise multiplication operation. As shown in Figure 3, each LSTM has a memory cell that comprises three gates (input, output, and forget) and a neuron with a self-recurrent connection.

The working steps of the LSTM are as follows:

- (i) The input gate allows incoming signal to block or alter the state of a memory cell.
- (ii) The forget gate decides what information to remove from the cell state.
- (iii) The $\tanh$ layer creates a vector of possible values to add to the cell state.
- (iv) Update the cell state by forgetting the old value and updating with new candidate values, scaled by how much updating is required for each state value.
- (v) Finally, the output value is decided based on the cell state.

The parameters of the LSTM layer are the dimensionality of the output layer (kept at 256), activation function ($\tanh$), the recurrent_activation function (hard_sigmoid), dropout as well as recurrent_dropout (kept at 0.2).

Figure 3: LSTM Memory Cell.

3.3 Convolutional Neural Network (CNN)

The convolutional neural network (<http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/>) is basically several layers of convolutions with nonlinear activation functions like *ReLU* or *tanh* followed by one or more fully connected layers. In a normal neural network, each input neuron is connected to each output neuron in the next layer, whereas in the CNNs, convolutions are used over the input layer to compute the output. Hence, each region of the input is connected to a neuron in the output.

In the CNN, the matrix represents sentences or documents passed as an input to the task. Each row of the matrix is a vector that represents a word. These vectors are usually word embeddings such as word2vec [28] or GloVe [31]. Filters are then used to slide over full rows of the matrix (words), and hence, “width” of filters is usually the same as the width of the input matrix. The *region size* or *height* may vary, but the sliding window size is of two to five words at a time usually. As shown in Figure 4, three filter region sizes (two to four) are chosen, and each has two filters. Each filter generates (variable-length) feature maps by performing convolutions on the sentence matrix. *1-max pooling* is then performed over each map to create a uni-variate

Figure 4: CNN Architecture.

feature vector from all six maps. A feature vector for the penultimate layer is formed by concatenating the six features. The final activation layer takes this feature vector as input and uses it to classify the sentence.

While building the CNN architecture, some of the parameters to choose from are input representations, i.e. embedding layer, number and size of convolution filters, pooling layer (*max*, *average*, etc.), and activation functions (*tanh*, *ReLU*, etc.). The convolution layer is the main building block of a CNN, which comprises a set of independent filters. The pooling layer reduces the number of parameters and computation in the network by progressively reducing the spatial size of the representation and operates on each feature map independently. After the activation function layer, the CNN usually has a fully connected layer at the end in which neurons in the layer have full connections to all activation in the previous layer. The CNNs are usually very fast and give reasonable results for the NLP tasks. The *1D Convolution* layer is used, which applies the convolution operator for filtering neighborhoods of one-dimensional input. The number of convolution kernels or dimensionality of output is set as 250, the *kernel_size* or the length of the *1D Convolution* window is kept at 3 with the activation function *ReLU* and a stride of 1. There was no padding applied to the input. The next layer is a *GlobalMaxPooling1D* layer, which does the max pooling operation. A dropout layer with a value of 0.2 is added to prevent over-fitting.

In each of the models discussed in Sections 3.3, 3.2, and 3.1, the penultimate layer is a *Dense* layer, which is a fully connected neural network with the output as the corresponding label of the review. The last layer of each model is an activation layer that applies an activation function (“*Sigmoid*” in the proposed models) to the output. After generation of the models, they are compiled using an optimizer (“*Adam*”, for this model), a loss function (*binary_crossentropy* for the implemented model), with an objective that the model tries to minimize. The optimizer “*adam*” uses 0.001 as the default learning rate. After compilation, the model is trained in a batch size of 32. Finally, the models are evaluated, batch by batch, for the loss and model accuracy.

4 Baseline Approach

The baseline approach is based on traditional machine learning techniques for comparing the results obtained using the proposed approach. One of the key ingredients for traditional machine learning techniques to work is the extraction of manually engineered features, on the basis of which the model tries to learn and predict class labels. Chen et al. [4] uses shallow syntactic features (bag of words, POS-n-gram, punctuation, hotel name), discourse parsing features, and sentiment features to build the classification model. The baseline model chosen to compare the proposed work implements the shallow features and sentiment features along with some more additional features such as the bag of uni-grams and bi-grams for more contextual information, length of review, and subjectivity of the review. The length of a review and subjectivity are chosen based on some heuristics, which suggest that truthful reviews are usually long and descriptive compared to the spam ones, which are comparatively more vague [21].

4.1 Features

- (i) Bag of words (F_1):
CountVectorizer [30] is used to convert each review into its corresponding vector for frequencies of each word in the review.
- (ii) Punctuation (F_2):
A binary feature to indicate a punctuation (‘?’ or ‘!’) in the review.
- (iii) POS-n-gram (F_3):
POS uni-gram and POS bi-gram tags are used as they help maintain the structure of the sentence.
- (iv) Bag of uni-grams and bi-grams (F_4):
Instead of the bag of words, the bag of uni-grams and bi-grams are considered as features. CountVectorizer [30] is used for generating the bag of uni-grams and bi-grams.

- (v) *Hotel_Name* (F_5):
This feature indicates whether *Hotel_Name* appears in the first line of the review.
- (vi) Length of the review (F_6):
An integer indicating the length (number of words) of each review.
- (vii) Subjectivity (F_7):
TextBlob [30] is used to calculate subjectivity of each review. It returns a float value [0.0, 1.0]: 0.0 means very objective, and 1.0 means very subjective.
- (viii) Sentiment words (F_8):
Three separate features corresponding to sentiment words are extracted: positive sentiment words, negative sentiment words, and total sentiment words. The average number of sentiment words used in the review is considered.
- (ix) Polarity of the review (F_9):
The range of the polarity is [0.0, 1.0] where 0.0 means the review is completely negative, and 1.0 means a highly positive review.

The feature vector is generated using the features (F_1 – F_9) extracted. Classification techniques such as Logistic Regression, Naive Bayes, and SVM are used to build the model.

5 Experiments and Results

In this section, the data set used for the experiments is described along with the results obtained. The baseline approach is used to evaluate and compare the results obtained by the proposed approach. All the proposed sequential models, as well as the model for the baseline approach, are coded in Python using machine learning libraries such as scikit-learn [30] and keras [5].

5.1 Data Set Description

The data set used for experimentation is the Deceptive Opinion Spam Corpus v1.4 [29]. The corpus contains 1600 truthful and deceptive hotel reviews of 20 Chicago hotels. The data set consists of the review, polarity of the review along with the label of the review (0 for not spam and 1 for spam). There are 400 deceptive and 400 truthful reviews of positive polarity. Similarly, there are 400 deceptive and 400 truthful reviews of negative polarity. The best state-of-the-art accuracy for the data set Deceptive Opinion Spam Corpus v1.4 is 89.5%, which is implemented by Chen et al. [4]. In the state-of-the-art model, the 400 reviews in each category are divided evenly into five-folds. Hence, each fold contains 80 reviews. A five-fold cross validation is then used for evaluation. The best model combines the bag of words, syntactic features, and discourse parsing features.

5.2 Evaluation and Discussion

This section discusses the evaluation of the proposed approach and its variants. The data set described in Section 5.1 has been split into an 80:20 ratio for training and testing, respectively. The split into train and test data set is done randomly in such a way that each set is representative of the whole data set. In order to regenerate the same results, seed is used.

5.2.1 Results for Baseline Approach

Features play an important role in traditional machine learning techniques, and hence, individual features are evaluated using the Logistic Regression model. It can be observed from Figure 5 that the bag of words

Figure 5: Results Obtained at the Feature Level.

feature yields a maximum accuracy of 86.25%, followed by *Hotel_Name* (64.58%). The rest of the features give accuracy in the range of 45–55% as depicted in Figure 5. The F-measure for the bag of words is 85.96%. The *Hotel_Name* feature gave an F-measure of 63.52%. The rest of the features yields a result in the interval of 43–60% for the F-measure. A similar trend is observed for precision. However, a slightly different trend observed for the recall values is shown in Figure 5. The maximum recall is obtained for the bag of words (87.83%). The positive polarity feature yielded the second best value of recall (82.61%) followed by the length of review (73.04%). The rest of the features are in the interval 40–65% for recall. Overall, the bag of words features obtained the best results using Logistic Regression. This may be due to the fact that the data set is engineered for spam detection and contains words that majorly contribute toward the improvement of the model.

The result for the combined feature vector that is generated using the individual features is evaluated using three different classifiers (SVM, Naive Bayes, and Logistic Regression) as shown in Figure 6. Logistic Regression gives the best results in terms of accuracy (87%), F-measure (86.58%), precision (86.2%), and recall (86.96%), followed closely by Naive Bayes, which gave an accuracy and F-measure of 73%, precision 70%, and recall 77%. The results obtained by Logistic Regression are better than the others because spam detection is a binary classification problem. The SVMs are usually more effective in the case of non-linear classification as they use kernel trick. Using SVM, the results are in the range of 53–56%.

5.2.2 Results for Proposed Approach

The accuracy obtained for the different variant sequential models based on the LSTM, RNN, and CNN are shown in Tables 1–3. Each model is evaluated using the different number of layers as a varying parameter.

Tables 1–3 discuss results obtained for the LSTM, RNN, and CNN, respectively, for the one, two, and three layers. In all the three cases, the best results are obtained in the case of a single-layer model as shown in Figure 7. The CNN gave an accuracy of 81%, and the LSTM gave 80% accuracy when a single layer is used, whereas the RNN gave an accuracy of 59% for a single layer. The layer-wise summary of the single-layer CNN model is shown in Table 4. The LSTM performs much better than the RNN because it considers long-term dependencies as well. The CNN helps in extracting position invariant features, whereas the RNN and LSTM are useful for modeling units in sequence. As the task at hand is to detect spam reviews, pattern detection

Figure 6: Comparison of the Different Techniques.

Table 1: Results for Different Models of LSTM.

Model	Accuracy	Precision	Recall	F-measure
1 Layer	0.79	0.77	0.8	0.78
2 Layers	0.74	0.71	0.79	0.75
3 Layers	0.73	0.68	0.87	0.76

Table 2: Results for Different Models of RNN.

Model	Accuracy	Precision	Recall	F-measure
1 Layer	0.59	0.57	0.62	0.59
2 Layers	0.53	0.5	0.59	0.54
3 Layers	0.61	0.58	0.63	0.6

Table 3: Results for Different Models of CNN.

Model	Accuracy	Precision	Recall	F-measure
1 Layer	0.81	0.78	0.84	0.81
2 Layers	0.7	0.66	0.8	0.72
3 Layers	0.69	0.65	0.76	0.7

is important. The LSTM preserves long-term structure, whereas the CNN is good at classification. Hence, the CNN gives slightly better results compared to the LSTM. The results show that increasing the number of layers of a particular model reduces the accuracy of the model in the case of spam detection.

While implementing a two-layer model where each layer is either a RNN, LSTM, or CNN, the best results are obtained in the model where the combination of LSTM and CNN is implemented. Table 5 shows the results for two-layered models, where 1 represents the first layer, 2 represents the second layer, and

Figure 7: Comparison of the Layered Models of the Different Architecture.

Table 4: Layer Wise Description of a Single-Layer CNN Model.

Layer (type)	Output shape	No. of parameters
embedding_1 (Embedding)	(None, none, 512)	10,240,000
spatial_dropout1d_1 (Spatial)	(None, none, 512)	0
conv1d_1 (Conv1D)	(None, none, 250)	384,250
global_max_pooling1d_1 (Glob)	(None, 250)	0
dense_1 (Dense)	(None, 250)	62,750
dropout_1 (Dropout)	(None, 250)	0
activation_1 (Activation)	(None, 250)	0
dense_2 (Dense)	(None, 1)	251
activation_2 (Activation)	(None, 1)	0

Table 5: Results for a Two-Layered Model.

LSTM	Simple RNN	CNN	Accuracy	Precision	Recall	F-measure
0	1	2	0.75	0.74	0.76	0.75
0	2	1	0.75	0.71	0.82	0.76
1	0	2	0.76	0.71	0.84	0.77
1	2	0	0.74	0.71	0.77	0.74
2	0	1	0.77	0.76	0.76	0.76
2	1	0	0.69	0.68	0.68	0.68

0 represents no layer. It can be observed that a change in the placement of a layer in the case of the LSTM and CNN models affected the results. The accuracy of the model with the first layer as the LSTM and the second layer as the CNN is 76.25%, whereas it is 76.67% when the first layer is the CNN layer and the second layer is the LSTM. The precision, recall, and F-measure for the LSTM–CNN network is 75.65% when the CNN is the first layer, whereas precision is 71.32%, recall 84.35%, and F-measure 77.29% when the LSTM is the first layer. The overall accuracy of the model is comparatively less when a layer of the RNN is introduced in the model because the RNN is a less-effective deep learning algorithm compared to the LSTM. The least accuracy is obtained in the combination of the RNN and LSTM (69% and 74%). The CNN is a better network architecture for text classification as discussed above, and hence, poor results are obtained when the CNN layer is missing in the architecture. When the first layer is LSTM, better results are obtained with a precision of 71%, recall of 76.52%, and F-measure of 73.64%. The LSTM performs better than the RNN because of long-term dependencies. As the second layer takes input from the first layer in a neural network, the more efficient the first layer is, the better is the feature extraction and, hence, improved results of the overall network.

In the case of the three-layered model variant, different combinations are built and evaluated as shown in Table 6, where the first layer represents that the particular model is used as the first layer of the sequential model. The second and third layers depict the respective positions of a layer in the model. The best result is obtained when the CNN is used as the first layer followed by the RNN and LSTM (78.75%). Further, an accuracy of 74.58% is obtained when the RNN is used as the first layer, followed by the LSTM and CNN. A common trend is observed that best results are obtained when the CNN is kept as a first layer. This is because the CNN performs the best feature extraction when compared to the LSTM or RNN. Moreover, the worst accuracy is obtained when the first layer is the LSTM, followed by the RNN and CNN layer (57.5%).

Output dimensions of the embedding layer is also varied to evaluate the effect on the accuracy. Best results are obtained in the proposed approach when the output dimensions are 512 (81.25%). As the number of dimensions increases in the embedding layer, the accuracy of the neural network increases, as shown in Table 7. The more the dimensions, the better is the representation of the reviews in vector space, and hence, the accuracy increases with an increase in the number of dimensions. However, the change in accuracy became near constant after 512 dimensions.

Figure 8 represents the t-SNE graph obtained by passing the penultimate layer of the single-layer CNN model to the t-SNE. The t-SNE graph shows two clusters, which clearly shows the distinct classification of the input reviews into spam and not-spam reviews. The receiver operating characteristic (ROC) curves and area under the curve (AUC) values for some of the best models in the proposed approach are shown in Figure 9. The AUC for the best model (single-layer CNN) as shown in Figure 9 is 0.8137.

5.2.3 Comparison of Proposed Approach with Baseline Approach

The baseline approach with the variant of Logistic Regression as a machine learning technique gave a maximum accuracy of 87% for the hotel reviews data set, whereas the proposed approach gave a maximum accuracy of 81.25%, which is comparable to traditional machine learning algorithms, if not better. The deep learning algorithms give much better results with a huge amount of data. As the data considered for all the above-obtained results is just 1600 reviews, which is very less for a deep learning model, the results obtained by the proposed approach did not surpass the baseline approach.

Table 6: Results for a Three-Layered Model.

LSTM	RNN	CNN	Accuracy	Precision	Recall	F-measure
1st Layer	2nd Layer	3rd Layer	0.575	0.5436	0.7044	0.6136
1st Layer	3rd Layer	2nd Layer	0.5792	0.5603	0.5652	0.5627
2nd Layer	1st Layer	3rd Layer	0.7167	0.688	0.7478	0.7167
2nd Layer	3rd Layer	1st Layer	0.7333	0.6946	0.7913	0.7398
3rd Layer	1st Layer	2nd Layer	0.5917	0.5714	0.5913	0.5812
3rd Layer	2nd Layer	1st Layer	0.7875	0.75	0.8348	0.7901

Table 7: Result for a Different Model of CNN.

Dimensions	Accuracy	Precision	Recall	F-measure
512	0.8125	0.7822	0.8435	0.8117
256	0.7917	0.748	0.8522	0.7967
128	0.7917	0.77	0.7913	0.7844
64	0.7792	0.7384	0.8347	0.7837
32	0.7625	0.7101	0.8522	0.7747
16	0.7083	0.6552	0.8261	0.7307

Figure 8: The t-SNE Graph Representing the Activation Values in a Single-layer CNN Model.

Figure 9: The ROC Curve for a Single-layer CNN.

6 Conclusion and Future Work

The proposed approach suggests that the deep learning sequential models perform comparably and are well suited to address the problem of review spam detection. A single-layer LSTM, CNN, or RNN model outperforms the multi-layer sequential models of the LSTM, CNN, or RNN. The single-layer LSTM model and the single-layer CNN model give 79% and 81.25% accuracy, respectively, whereas the single-layer RNN model gives an accuracy of 59%. When a combination of the LSTM, CNN, or RNN is used in a two-layered sequential model, the CNN–LSTM combination gives the best results (77% accuracy). A 78.75% accuracy was obtained when the first layer of a three-layered sequential model implemented the CNN, the second layer RNN, followed by the LSTM in the last layer. For the CNN model, the results have improved with an increase in the number of output dimensions of the embedding layer in the deep network model. Larger networks having more number of dimensions help capture information better and, hence, yield better results. An accuracy of 81.25% was obtained when the number of output dimensions of the embedding layer was 512. A comparison is drawn between the traditional machine learning algorithms and deep learning algorithms for spam review classification. The baseline approach resulted in an accuracy of 87% using Logistic Regression. The results show

that the traditional machine learning model gives a higher accuracy for small-sized data. However, the deep learning models are not far behind. With larger data sets, deep learning techniques are likely to surpass traditional machine learning algorithms. In future work, the aim is to implement these sequential deep learning models on larger data sets and then compare the performance.

Bibliography

- [1] F. Ahmed and M. Abulaish, An MCL-based approach for spam profile detection in online social networks, in: *11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 602–608, IEEE, Liverpool, UK, 2012.
- [2] S. P. Algur, A. P. Patil, P. S. Hiremath and S. Shivashankar, Web based customer review spam detection using conceptual level similarity measure and shingling technique, *IJIRST* (2011), 1–9.
- [3] M. Chakraborty, S. Pal, R. Pramanik and C. R. Chowdary, Recent developments in social spam detection and combating techniques: a survey, *Inf. Process. Manag.* **52** (2016), 1053–1073.
- [4] C. Chen, H. Zhao and Y. Yang, *Deceptive opinion spam detection using deep level linguistic features*, Natural Language Processing and Chinese Computing, pp. 465–474, Springer, 2015.
- [5] F. Chollet et al., *Keras*, <https://github.com/fchollet/keras>, 2015.
- [6] H. Costa, F. Benevenuto and L. H. C. Merschmann, Detecting tip spam in location-based social networks, in: *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pp. 724–729, ACM, New York, NY, USA, 2013.
- [7] K. Dave, S. Lawrence and D. M. Pennock, Mining the peanut gallery: opinion extraction and semantic classification of product reviews, in: *Proceedings of the 12th International Conference on World Wide Web*, pp. 519–528, ACM, New York, NY, USA, 2003.
- [8] M. Diale, C. Van Der Walt, T. Celik and A. Modupe, Feature selection and support vector machine hyper-parameter optimisation for spam detection, in: *Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech)*, 2016, pp. 1–7, IEEE, Stellenbosch, South Africa, 2016.
- [9] S. Fakhraei, J. Foulds, M. Shashanka and L. Getoor, Collective spammer detection in evolving multi-relational social networks, in: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1769–1778, ACM, Sydney, NSW, Australia, 2015.
- [10] G. Fei, A. Mukherjee, B. Liu, M. Hsu, M. Castellanos and R. Ghosh, Exploiting burstiness in reviews for review spammer detection, in: *ICWSM*, pp. 175–184, Boston, USA, 2013.
- [11] S. Feng, L. Xing, A. Gogar and Y. Choi, Distributional footprints of deceptive product reviews. in: *ICWSM*, pp. 98–105, Dublin, Ireland, 2012.
- [12] H. Fu, X. Xie and Y. Rui, Leveraging careful microblog users for spammer detection, in: *Proceedings of the 24th International Conference on World Wide Web*, pp. 419–429, ACM, Florence, Italy, 2015.
- [13] X. Hu, J. Tang, Y. Zhang and H. Liu, Social spammer detection in microblogging, in: *Twenty-Third International Joint Conference on Artificial Intelligence*, pp. 2633–2639, Beijing, China, 2013.
- [14] E. Huang, Paraphrase detection using recursive autoencoder, *Source*: [<http://nlpstanford.edu/courses/cs224n/2011/reports/ehhuang.pdf>] (2011).
- [15] S. Jeong, G. Noh, H. Oh and C.-K. Kim, Follow spam detection based on cascaded social information, *Inf. Sci.* **369** (2016), 481–499.
- [16] N. Jindal and B. Liu, Opinion spam and analysis, in: *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pp. 219–230, ACM, Palo Alto, CA, USA, 2008.
- [17] A. Kantchelian, J. Ma, L. Huang, S. Afroz, A. Joseph and J. D. Tygar, Robust detection of comment spam using entropy rate, in: *Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence*, pp. 59–70, ACM, Raleigh, NC, USA, 2012.
- [18] I. Kayes, N. Kourtellis, D. Quercia, A. Iamnitchi and F. Bonchi, The social world of content abusers in community question answering, in: *Proceedings of the 24th International Conference on World Wide Web*, pp. 570–580, ACM, Florence, Italy, 2015.
- [19] J. M. Kim, Z. M. Kim and K. Kim, An approach to spam comment detection through domain-independent features, in: *Big Data and Smart Computing (BigComp)*, 2016 International Conference on, pp. 273–276, IEEE, Hong Kong, China, 2016.
- [20] K. Lee, B. David Eoff and J. Caverlee, Seven months with the devils: a long-term study of content polluters on Twitter, in: *ICWSM*, pp. 185–192, Barcelona, Spain, 2011.
- [21] J. Li, M. Ott, C. Cardie and E. H. Hovy, Towards a general rule for identifying deceptive opinion spam, in: *ACL (1)*, pp. 1566–1576, Citeseer, 2014.
- [22] H. Li, Z. Chen, A. Mukherjee, B. Liu and J. Shao, Analyzing and detecting opinion spam on a large-scale dataset via temporal and spatial patterns, in: *ICWSM*, pp. 634–637, Oxford, UK, 2015.
- [23] H. Li, G. Fei, S. Wang, B. Liu, W. Shao, A. Mukherjee and J. Shao, Bimodal distribution and co-bursting in review spam detection, in: *Proceedings of the 26th International Conference on World Wide Web*, pp. 1063–1072, International World Wide Web Conferences Steering Committee, Perth, Australia, 2017.

- [24] L. Li, B. Qin, W. Ren and T. Liu, Document representation and feature combination for deceptive spam review detection, *Neurocomputing* **254** (2017), 33–41.
- [25] Y. Lin, T. Zhu, H. Wu, J. Zhang, X. Wang and A. Zhou, Towards online anti-opinion spam: spotting fake reviews from the review sequence, in: *Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on*, pp. 261–264, IEEE, Beijing, China, 2014.
- [26] C. Lumezanu and N. Feamster, Observing common spam in Twitter and email, in: *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, pp. 461–466, ACM, Boston, MA, USA, 2012.
- [27] G. Mi, Y. Gao and Y. Tan, Apply stacked auto-encoder to spam detection, in: *International Conference in Swarm Intelligence*, pp. 3–15, Springer, 2015.
- [28] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado and J. Dean, Distributed representations of words and phrases and their compositionality, in: *Advances in Neural Information Processing Systems*, pp. 3111–3119, 2013.
- [29] M. Ott, Y. Choi, C. Cardie and 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*, pp. 309–319, Association for Computational Linguistics, Portland, Oregon, 2011.
- [30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, Scikit-learn: machine learning in Python, *J. Mach. Learn. Res.* **12** (2011), 2825–2830.
- [31] J. Pennington, R. Socher and C. D. Manning, Glove: Global Vectors for Word Representation, in: *EMNLP*, 14, pp. 1532–1543, Doha, Qatar, 2014.
- [32] M. Poorgholami, M. Jalali, S. Rahati and T. Asgari, Spam detection in social bookmarking websites, in: *Software Engineering and Service Science (ICSESS), 2013 4th IEEE International Conference on*, pp. 56–59, IEEE, 2013.
- [33] A. Popov, Deep learning architecture for part-of-speech tagging with word and suffix embeddings, in: *International Conference on Artificial Intelligence: Methodology, Systems, and Applications*, pp. 68–77, Springer, Varna, Bulgaria, 2016.
- [34] C. Rădulescu, M. Dinsoreanu and R. Potolea, Identification of spam comments using natural language processing techniques, in: *Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on*, pp. 29–35, IEEE, Cluj-Napoca, Romania, 2014.
- [35] Y. Shao, M. Trovati, Q. Shi, O. Angelopoulou, E. Asimakopoulou and N. Bessis, A hybrid spam detection method based on unstructured datasets, *Soft Computing* **21** (2017), 233–243.
- [36] M. Sirivianos, K. Kim and X. Yang, SocialFilter: introducing social trust to collaborative spam mitigation, in: *INFOCOM, 2011 Proceedings IEEE*, pp. 2300–2308, IEEE, Shanghai, 2011.
- [37] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng and C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank, in: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, 2013.
- [38] L. Song, R. Y. K. Lau, R. C.-W. Kwok, K. Mirkovski and W. Dou, Who are the spoilers in social media marketing? Incremental learning of latent semantics for social spam detection, *Electron. Commer. Res.* **17** (2017), 51–81.
- [39] I. Sutskever, O. Vinyals and Q. V. Le, Sequence to sequence learning with neural networks, in: *Advances in Neural Information Processing Systems*, pp. 3104–3112, Palais des Congrès de Montréal, Montréal, Canada, 2014.