Detection of Counterfeit News Using Machine Learning

Anjali B, Reshma R, Geetha Lekshmy V
Department of computer science and applications
Amrita vishwa vidyapeetham, Amritapuri,
India
anjaliusha29@gmail.com, reshmaram008@gmail.com, geethalekshmy@am.amrita.edu,

***Abstract*—Counterfeit news or hoax news has been one of the prevalent problems that global society has faced. The differentiation between counterfeit news and real news has stayed one of the hardest problems to tackle. The proposed system comes up with models of different machine learning techniques for differentiating fake news and real news. This approach utilizes four different models of machine learning: Neural Network, Naive Bayesian, Support Vector Machine (SVM) and Long and Short-Term Memory(LSTM). The dataset that we are using is obtained from Kaggle(Fake News). We start with text normalization of the data(news) in the dataset. Text normalization includes stopping words removing, removing punctuations and other diacritics, converting all letters to lowercase, etc. The second step is feature extraction. Doc2vec model is used in order to represent the data as feature vectors. These vectors are the inputs of Naive Bayes, SVM, Neural network and LSTM. Every one of these algorithms will be assessed to determine which algorithm will yield the most accurate result. Among the four algorithms, LSTM proved to be the best with an accuracy of (92%).**

***Index Terms*—Doc2Vec, Feature Vector, Naive Bayesian, Support Vector Machine, Neural Network, Long Short Term Memory**

I. Introduction

Pseudo-news or fake news is exploitative journalism which is misleading and spreads via media. The objective of this propaganda is to earn public attention to increase readership or to suppress a person or an organization purposefully for someone's sake. Spreading of pseudo-news makes it more difficult to differentiate benign news from fake news and most importantly it compromises social media coverage. To solve these problems, it is essential to identify hoax news from amongst real news. This paper aims to explore various techniques that can be employed in order to identify if a given news extract is real or not. Our goal is to determine the best classification algorithm to determine fake news and real news. For that, a data set of news which consists of data until 2014 is used. The various steps involved in the process are:

- Text cleaning
- Feature extraction using Doc2Vec model.
- Build classification models using Naive Bayesian, Support Vector Machine, Neural Network, Long Short-Term memory.
- Find the most accurate algorithm for counterfeit news detection.

II. Related Works

T. Mikolov and K. Chen in their paper[9] proposes how to compute the continuous representation of vectors of each word using two novel architecture. We observed that models like Doc2Vec can be used to train high-quality word vectors. Machine learning algorithms require feature vectors as inputs. One of the most common fixed length features is bag-of-words features. But it loses ordering and semantics of words. In this paper[5], an unsupervised algorithm called paragraph vector is proposed. Doc2Vec uses skip-gram which consider both semantics and ordering of words. Therefore, in this project, we are using Doc2Vec to create document vectors.

In paper[6] authored by M.Granik and V.Mesyura, they describe rather a basic approach using Naive Bayes classifier. This method was tested against a dataset of a well-known social networking site's news.

In their paper[7] S.Aphiwongsophon and P.Chongstitvatana use three machine learning techniques namely Neural Network, Naive Bayes and Support Vector Machine for classification of fake and real news. They found that Support Vector Machine and Neural Network yield better result that Naive Bayes.

N. J.Conroy, V.L. Rubin et al. in their paper[8] uses a hybrid approach for fake news detection. It uses linguistic approach, that analyses the content of the news data, and network approach that uses meta information of the news, to detect fake news.

Y.Long, Q.Lu, R.Xiang et al. in their paper[10] proposed a method for fake news detection. In this method, an LSTM model which is attention-based is utilized to subsume a speaker profile. In this paper, they are using two LSTM's and concatenating the result to get the actual result.

In the paper[12] by S.Chopra, S. Jain et al. Support Vector Machine based on TF-IDF and cosine similarity measures is used to classify headline article as related or unrelated. Related ones are classified using Long Short Term Memory Model and Bidirectional Conditionally Encoded LSTMs with Bidirectional Global Attention proved to be the best model.

III. Methodology

A. Doc2Vec

The most challenging part of applying Machine Learning is the representation of textual data in numerical form. Nu-

meric representation of text document can be used for document retrieval, web searching, topic modeling, spam filtering, etc.Based on the former Word2Vec model[9], a new model called Doc2Vec was formulated by Thomas Mikilov and Le[5] in 2014. Word2Vec is a well-known concept used to represent words as vectors. The Word2vec presented in 2013 intends to give a numeric representation of each word that will be able to capture relations between them. Word2Vec representation uses two algorithms for its working: skip-gram model and continuous bags-of-words model[9]. CBOW predicts from "context" by utilizing a sliding window around the given word. Every word obtained this way is represented as a feature vector. Consequently, word vectors are obtained by training these vectors. Skip-gram[9] works in contrast with the CBOW method. Skip-gram basically utilizes one given word to predict the surrounding word unlike CBOW, which uses one word each time. Skip-gram is more accurate and much slower than CBOW. However, word documents lack logical structure and thus, the method is inadequate for the purpose. Word2Vec model consists of another feature vector, which is unique to a document. At the end of the training, the document vector x and word vector w are trained. The vector x is a vector that numerically represents a document. The result that obtains is a model called the distributed memory of paragraph vector.

B. Naive Bayesian

The root of Naive Bayesian algorithm[4] is Bayes theorem[6]. It is a simple algorithm for predictive analysis. it is easy to build, especially for large dataset . Bayes theorem is a method to find out the conditional probability. The probability of an event happening even that it has some relation among other events is called conditional probability. The definition of the Bayes theorem is given a hypothesis C and evidence X, Bayes theorem states that the relation between the probability of the hypothesis before getting the evidence P(C) and the probability of the hypothesis after getting the evidence.

$$P(H|D) = \frac{P(D|H)P(H)}{P(D} \tag{1}$$

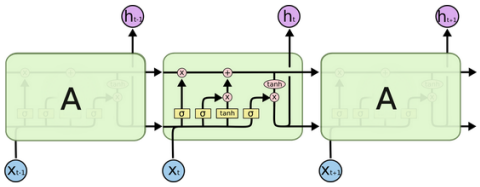
When you fit your design matrix and target values to the Naive Bayes model you fit the model according to the design matrix and target values. Design matrix and target values are the training sets. In that set, each example is a pair of input object which is a vector and the desired output value which is a supervisory signal. Design_matrix is a matrix with training examples with dimension m x n, where m is the number of examples and n is the number of features.Target_values are the values with dimension m. The algorithm will extract the features of each record in the training examples. The features may be continuous or discrete. Therefore if it is continuous the algorithm converts the features into continuous feature vector else convert it into a discrete feature vector. The algorithm iterates the target_value and updates the prior list with the mean of each element in the target_value. So after training the model now, it contains the mean variance of each label. This mode is further used for the prediction of the test data or any other dataset of news with the same features. The test data is a dataset with dimension m x n. To predict the target_value of the test_set you need to pass the test_set to the trained model using predict function. The function returns the predicted target values for the test set with dimension m. The algorithm extracts the features of each record in the test data. These features are used to find out the probability of each label for the particular record. The likelihood of the occurrences are obtained and their corresponding log values are stored in a dictionary.

log_likelihood = k: log(v) for k, v in self.priors.items()
The key is k, which is the log of value in the prior dictionary.

After checking the probability in test_records the algorithm will find the log of the probable value and then return the max value out of the log_likelihood. This is probably the possible match found from the test data being returned. The label which has the maximum value out of the log_likelihood is the label of the corresponding record.

C. LSTM

LSTM is a type of Recurrent neural network which is a kind of artificial neural network. It is designed to identify the patterns in a sequence of data, such as data generating from sensors, genomes, stock market, handwriting, numerical times series, etc. RNN has new information along with information from the previous timestamp. The output that we got in the previous timestamp use certain information from that and feed it to the network as input and that will help us to get the new output similarity. The new output that we have to get take some information from that find it as an input to the network along with the new information to get the new prediction and it keeps repeating. LSTM is capable of learning long-term dependencies. We have to refer the previous information so that we can perform the present task[10].



The repeating module in an LSTM contains four interacting layers.

In standard RNN the module which is repeating is a simple solution such as a single tanh layer[10]. This tanh layer is nothing but a squashing function to convert the value between -1 and 1. LSTM has a structure like a chain and a module that is repreating has a different structure.There are four Neural Network Layers in LSTM[3]. In a special way they are interacting. Let x be a set of news and y be the collection of authors. A sample news decay element of x includes text as a sentence z_k , and an author $y_k \epsilon y.A\ sentence$ z_k is formed by a sequence of words $z_k = w_{k1}, w_{k2}, ..., w_{kl}$ Where l is the length of z_k . The word features $w_i \epsilon z_k$ from a word vector w_i with length l. Vector $w_i = f_1 w_i, f_2 w_i, ..., f_l w_i$. Every $w_k \epsilon f_k$ and $y_k \epsilon$ y are hand over to the first LSTM. The author,

vector y and the title vector t are the attention factors for z_k. z_k and t_k are the output of these LSTM models. z_k and y_k are concatenated to form x_k .The final layer project x_k on to the target space of L class labels through a softmax layer. https://www.overleaf.com/project/5cce5e0f558fac6ebf1adef1

To remove the not required information from the self-state is the initial step in LSTM. A layer called forget gate layer Fi which is a sigmoid layer makes the decision. The result will be a number ranging from 0 and 1. For each number that coming from the previous timestamp in the cell state Ht-1. Zero means completely keep. one means completely get rid off.

$$f_t = \alpha(w_f(h_{zk:wi} - 1, \vec{w}_i) + b) \tag{2}$$

The new information that must be stored in the self-state is decided in the next step. Input gate layer decides the value that must be updated and the vectors of new candidate values are created by the tanh layer. These new candidate values are added to the state.

$$c_t = tanh(w_c(h_{zk:wi} - 1, \vec{w}_i) + b) \tag{3}$$

$$i_t = \alpha(w_c(h_{zk:wi} - 1, \vec{w}_i) + b) \tag{4}$$

Now we will update the cell state c_{t-1} into c_t . First, multiple c_{t-1} by ft. Add i_t and c~t to generate the new candidate value

$$c_t = f_t \times c_{t-1} + i_t \times c^{\sim}t \tag{5}$$

A sigmoid layer decides what part of cell state that going to output. Put the cell state through tan and multiply by the output of the sigmoid state.

$$h_{zk:wi} = o_i tanh(f_i, c_i) \tag{6}$$

The attention mechanism employs a weighting scheme.This is done so as to designate how crucial different words are in identifying the meaning of a given news article. The author and title for training weight are represented as continuous and real valued vectors z and t. Let α w$_i$ denote the weight to estimate the relevance and significance of w_i with respect to z_kThe attention weight $\alpha(foreverystate)definedas$

$$\alpha w_i = \frac{exp(e(h_{zk:wi}\vec{z}, \vec{t}))}{\sum_{k=1}^{l_i} exp(e(h_{zk:wi}\vec{z}, \vec{t}))} \tag{7}$$

The importance of words indicated by the score function e.

$$exp(e(h_{zk:wi}, \vec{z}, \vec{t})) = \upsilon^T tanh(W_H h_{zk:Wi} + W_z\vec{z} + W_t\vec{t} + b) \tag{8}$$

The enhanced sentence representation Vector z_k is added with weights in some of the stages that are not visible.

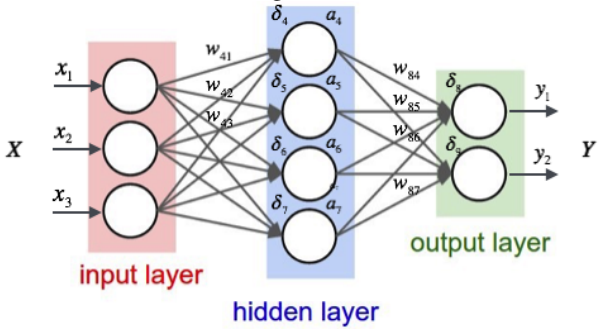
$$\vec{z}_k = \sum \alpha w_i h_{zk:wi} \tag{9}$$

Similarly, second LSTM model represents the author by concatenating y_k and z_k to compute x_k , representation of final news.

$$\vec{x}_k = \vec{z}_k \oplus \vec{y}_k \tag{10}$$

D. Neural Network

Neural networks[2] are a circuit of neurons or a network which is commonly designed to identify patterns. They recognize patterns that are numerical, images, text, or in vector form. Neural networks learn by example. It helps us to cluster or classify unlabelled data according to a labeled dataset. The classification process mostly depends on the dataset that is labeled so that it helps a neural network to learn the connection between data and labels. It consists of components such as a set of weights(h), an input layer(i), biases(b) and an output layer(j) between each layer, an arbitrary amount of hidden layers and for each hidden layer, there will be a choice of activation function called sigmoid.



An architectural diagram for Neural Network

In a 2-layer Neural Network, the output $\bar{j}$ can be predicted by using the given equation.

$$\bar{j} = \sigma(h_2\sigma(h_1 x + b_1) + b_2) \tag{11}$$

Here the only variables such as biases(b) and weights(h) will affect the output $\bar{j}$. ie; the strength of the predictions strongly depends upon when the biases and weights have the right values. Minute essential alterations are made to weights and biases being added, in the form of input. This process is called Training the Neural Network and is done so as to enhance the output results. The two iterations in the training process are feedforward (calculating the output prediction j) and backpropagation (updating the biases and weights). The feedforward can be calculated by using the same equation(11) for a fundamental 2-layer neural network. The loss function will help us to check the goodness of our predictions by using the equation(12).

$$Sum - of - SquaresError = \sum_{i=1}^{n}(j - \bar{j})^2 \tag{12}$$

By adding up the difference between each actual value and predicted value, the sum of the square error is derived.

Squaring of the difference gives the accurate difference value. The weights and biases are subjected to updation by disseminating the errors back. This is called backpropagation. The derivative of the loss function pertaining to biases and weights are considered when regulating the biases and weights. To calculate the loss functions derivative, the below equation(13) can be used.

$$loss(j, \overline{j}) = \sum_{i=1}^{n} (j - \overline{j})^2 \tag{13}$$

E. Support Vector Machine

The main goal of this method is to find out a hyperplane that classifies the data points in an N-dimensional space. This line is considered as the decision boundary. There are many possible hyperplanes that could be chosen for the separation of data points belonging to two different classes. So, the prime intention here is to locate a plane with the largest distance/margin between identified data points of the two different classes. A margin is a space that exists between the two lines on the closest class points, which is calculated as the perpendicular distance from the line to support vectors or closest points. A margin is said to be a good margin if the margin is wider in between the classes. The hyperplanes dimension is a factor that relies upon the number of input features. If there are two features, the hyperplane defines a line and if there are three features, the hyperplane exhibits on a three-dimensional plane. Support vector is the data points which is at the least minimal distance to the hyperplane. Thus it also contributes in defining the orientation and position of the hyperplane. Thereby, it is clear that the removal of any given support vector may generate a change in the position of the hyperplane. It can be also used to maximize the margin of the classifier and these points help to build the SVM model. The SVM[1] is implemented by using a kernel, which changes an input data space into the required form. Kernels are of various types: Polynomial Kernel, Radial Basis Function kernel, Linear kernel. The linear kernel is the dot product of any two given observations whereas the product between the two vectors is the sum of the multiplication of each pair of the input values.

$$K(q, qi) = sum(q * qi) \tag{14}$$

Polynomial Kernel can manage nonlinear or curved input space, where d is the degree of the polynomial.

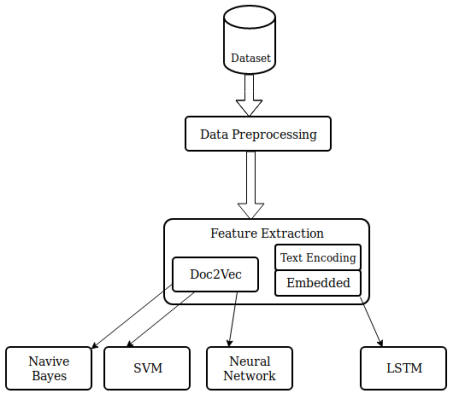
$$K(q, qi) = 1 + sum(q * qi)^d \tag{15}$$

Radial Basis Function Kernel can map an input space in infinite dimensional space, where gamma ranges from 0 to 1.

$$K(q, qi) = exp(-gamma * sum((q - qi^2)) \tag{16}$$

The implementation of the SVM requires a dataset which comprises of features(text/attributes) and target type(label).

IV. Proposed Solution



The dataset is from Kaggle. It consists of 17500 rows of data. The attributes in this dataset are a unique id for a news article, the title of the news a, author of the news, the text of the news and a label that mark news as reliable or unreliable. News is marked as 1 is reliable and unreliable news are marked as 0. After obtaining the data, text normalization will be done. Text normalization includes converting letters to lower case, remove punctuation and accent marks, remove white space and stop words, etc. we are using doc2vec to represent news as its feature vector. The vector of a document represents its concept. Therefore by seeing a vector, the machine will understand the meaning and its semantic relation with other sentences. These vectors are stored in numpy arrays. The size of these arrays is exact to the size of the trained dataset and test dataset. Ie, two numpy arrays are for trained data and its labels and the next two numpy arrays are for test data and its labels. These arrays are the inputs for each algorithm.

When you use an algorithm, you need to train the dataset with that particular algorithm to create a model. Using this model the labels of test data can predict . Since words order is important in LSTM, Doc2Vec cant use to create the feature vector. Because it transfers the document into one vector and loses the order of information. Therefore word embedded is used in LSTM instead of Doc2vec.

V. Results and Analysis

For evaluating different classification algorithm's performance the following metrics accuracy,recall, precision, and F1 score are used.

Before accuracy, F1-score, precision, and recall, we need to know what is True Positive, True Negative, False Positive and False Negative. The real news that is correctly predicted by the algorithm is called True positives. True Negatives are the fake news that the algorithm correctly predicted. False Positives actual class is fake and predicted class is real. When actual class is real but the algorithm predicted it as fake can call it as False negatives. These four parameters used to calculate F1 score, Accuracy, Recall, and Precision.

Accuracy is a ratio correctly predicted observation to the total observations. A model with high accuracy is said to be the best model

$$Accuracy = TP + \frac{TN}{TP} + FP + FN + TN \tag{17}$$

Precision is defined as the ratio of predicted positive observations that are correct to the total predicted positive observations.

$$Precision = \frac{TP}{TP + FP} \tag{18}$$

The recall is defined as the ratio of correctly predicted positive observations to all observations in actual class - yes.

$$Recall = \frac{TP}{TP + FN} \tag{19}$$

F1 Score is the weighted average of both Precision and Recall. As a result, this score considers both false negatives and false positives.

$$F1Score = \frac{2 \times (Recall \times Precision)}{(Recall + Precision)} \tag{20}$$

TABLE I

Algorithm performance				
Algorithms	Accuracy	Precision	Recall	F1-Score
Naive Bayes	0.72	0.67	0.85	0.75
SVM	0.88	0.84	0.93	0.88
Neural Network	0.90	0.86	0.94	0.89
LSTM	0.92	0.88	0.95	0.91

On comparing these algorithms, Naive Bayes yields the poor results (Accuracy- 72%, Precision-67%, Recall-85%, F1-Score-75%).SVM yields somewhat better results than Naive Bayes having Accuracy-88%, Precision-84%, Recall-93%, and F1-Score-88%. Both LSTM and Neural Networks have almost similar results. But, LSTM is more accurate than Neural Network by achieving Accuracy-92%, Precision-88%, Recall-95%, F1-Score-91% whereas Neural Network has Accuracy-90%, Precision-86%, Recall-94%, F1-Score-89%.

VI. Conclusion

Among the four algorithms, LSTM gave the best results. the highest F1 score is achieved by LSTM comparing to all the other models. The LSTM performs well because the text is inherently a serialized object. Other models use Doc2Vec to get their feature vectors. But the LSTM model preserves the order and make a prediction based on both words and their orders. Therefore the most accurate algorithm for classification is LSTM.

References

[1] A. Baskar and T. G. Kumar, "Facial expression classification using machine learning approach: A review," in *Data Engineering and Intelligent Computing*. Springer, 2018, pp. 337–345.

[2] S. Lalitha, S. Tripathi, and D. Gupta, "Enhanced speech emotion detection using deep neural networks," *International Journal of Speech Technology*, pp. 1–14, 2018.

[3] R. Vinayakumar, K. Soman, P. Poornachandran, and S. Sachin Kumar, "Detecting android malware using long short-term memory (lstm)," *Journal of Intelligent & Fuzzy Systems*, vol. 34, no. 3, pp. 1277–1288, 2018.

[4] S. Sathya, S. Joshi, and S. Padmavathi, "Classification of breast cancer dataset by different classification algorithms," in *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*. IEEE, 2017, pp. 1–4.

[5] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, 2014, pp. 1188–1196.

[6] M. Granik and V. Mesyura, "Fake news detection using naive bayes classifier," in *2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON)*. IEEE, 2017, pp. 900–903.

[7] S. Aphiwongsophon and P. Chongstitvatana, "Detecting fake news with machine learning method," in *2018 15th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*. IEEE, 2018, pp. 528–531.

[8] N. J. Conroy, V. L. Rubin, and Y. Chen, "Automatic deception detection: Methods for finding fake news," *Proceedings of the Association for Information Science and Technology*, vol. 52, no. 1, pp. 1–4, 2015.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[10] Y. Long, Q. Lu, R. Xiang, M. Li, and C.-R. Huang, "Fake news detection through multi-perspective speaker profiles," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 2017, pp. 252–256.

[11] A. Roy, K. Basak, A. Ekbal, and P. Bhattacharyya, "A deep ensemble framework for fake news detection and classification," *arXiv preprint arXiv:1811.04670*, 2018.

[12] S. Chopra, S. Jain, and J. M. Sholar, "Towards automatic identification of fake news: Headline-article stance detection with lstm attention models," 2017.

[13] F. Qian, C. Gong, K. Sharma, and Y. Liu, "Neural user response generator: Fake news detection with collective user intelligence." in *IJCAI*, 2018, pp. 3834–3840.