Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Rumor detection in Twitter: Determining Useful Attributes for classifications

First A. Author1, Fellow, IEEE, Second B. Author2, and Third C. Author, Jr.3, Member, IEEE

1National Institute of Standards and Technology, Boulder, CO 80305 the USA

2Department of Physics, Colorado State University, Fort Collins, CO 80523 USA

3Electrical Engineering Department, University of Colorado, Boulder, CO 80309 the USA

Corresponding author: First A. Author (e-mail: author@ boulder.nist.gov).

This paragraph of the first footnote will contain support information, including sponsor and financial support acknowledgment. For example, “This work was supported in part by the U.S. Depart­ment of Com­merce under Grant BS123456.”

ABSTRACT With the growth of social media such as Twitter in the community, it became easier for people to get data or spread rumors. But developing an automatic moderation system to check the credibility of the data is a challenging work, an alternate approach is with the help of users or third-party organization to flag those tweets which are suspected to be rumors, but this method is highly inefficient and redundant, hence making our social media an unreliable source for Information. As a part of this work, we aim to develop an algorithm to segregate such tweets and flag them as rumors. We used three publicly available twitter datasets to find such discriminative features which can classify a tweet as a rumor or non-rumor. Using two recurrent (bi-directional LSTM) neural models and two artificial neural models, the results in the work provide us evident information to correlate important features for classifying the tweets. Along with the result, we have also proposed methods for attributes extraction from user and tweets and pre-processing techniques to retain the key information.

INDEX TERMS Twitter 15 dataset, Twitter 16 dataset, Tweet meta-data attributes, PHEME dataset, Rumor detection, User meta-data extraction.

I. INTRODUCTION

As per Peterson and Warren, a rumor is defined as “a tall tale of explanation of events circulating from person to person and about an object, event, or issue in public concern” [1]. In layman's terms rumor is a statement whose veracity is not confirmed quickly or at all, but it is propagating among a bunch of people. Rumors give birth to the spread of misinformation (false information) or disinformation (deliberate false information) in public. Rumors have been a part of society from historical times. With the birth of new-age technologies and the development of social media sites, the grounds for connectivity have rapidly increased. Among these sites, Twitter has emerged as a leading source of information sharing, where, on average 500 million tweets are shared per day and each user, by design, can tweet up to 140 characters long messages or share files like pictures or videos. Twitter supports a follower and following design in which a user can subscribe to some other user’s tweets to receive information posted by the latter. Such a system provides a fluent flow of data. These tweets sometimes contain resourceful information like public announcements by governing organizations, sharing of events, personal experiences, beliefs, thoughts, and many more. Camouflaging among this data, rumors are widely spreading and destroying the credibility of this information. This problem is elevated due to crowdsourcing where a rumor can be backed by many users due to a lack of knowledge regarding the matter and their beliefs. To tackle this issue a lot of work has been done in the research community to find a prominent solution. With this work, we try to understand the root cause of how a rumor is generated on Twitter, its varying nature, way of propagation, the targeted audience, and attributes through which it is easier to determine whether an online post is a rumor or not. Note: the research doesn’t include finding the veracity of the rumor tweet, its main purpose is rumor classification only.

We propose three neural network-based solutions to understand the insight in tweet data and to map a trend for the classification task. All the models are developed as a supervised learning task on twitter’s publicly available dataset from which tweet-id and labels are gathered and then using respective tweet-id, tweets are collected in JSON format using Twitter API. Labels in the dataset are categorized into three categories as follows:

* Non-rumor
* True (rumor whose veracity is defined as correct information on a later period, but it started spreading as rumor initially.)
* False (rumor whose veracity is defined as incorrect information or deliberate misinformation after extensive search either from a third-party organization like PolitiFact or gossipcon or by flagging such tweets with the help of users.)
* Un-verified (are those rumor spreading tweets whose veracity is still questionable and people are still verifying their states.)

To further investigate the working and to develop these models, a basic understanding of dataset labels is necessary. A general misconception in masses is that a piece of fake news or a rumor, categorized as false, is the only rumor data and to verify this hypothesis we developed a recurrent neural model to find the probability of the tweet text content being fake and to understand a relation. This model is also developed as a supervised learning task on FakeNewsNet [2] dataset, trained on nearly 50,000 tweets. General overviews of the solutions are:

1. User and Tweet meta-data model: is an artificial neural network model which takes users and tweet meta-data and learns the discriminative attributes for the classification task.
2. Tweet text analysis model: is a recurrent neural network model which performs NLP task on tweets text content to understand relevant information for classification.
3. Reaction on comment model: which is a sentiment analyzer to understand the average sentiment of the tweet comments and then using it as an attribute for rumor classification.
4. Fake tweet classification: is also a recurrent neural network model to get the probability of the text being fake or real.

After this, all models were ensembled together and trained on the dataset to get the final weights of all the models. These weights determine how much the rumor classification task is dependent on the models which in turn provides the weight to understand the importance of different tweet attributes. Overall the contributions are mainly in eight aspects.

1. Proposing a feature extraction mechanism and preprocessing techniques to make data more relevant.
2. Developing a hypothesis on metadata in user and tweet dataset and map related attributes to the problem.
3. Developing two NLP models to understand the text features, emotion, POS, general topic, and other attributes for rumor and fake tweet classification.
4. Correlating derived and extracted feature to realize the relationship between attributes and reduce the dimensionality of efficient resource utilization.
5. Ensembling techniques for combining the models and training and testing the final model for a general solution.
6. Understanding the weight distribution of feature variables and formulating their importance.
7. Defining loss metrics for the classification task on the dichotomous rumor data.

II. RELATED WORKS

In this section, the background research and studies conducted by us to understand the problems still persistent in rumor studies and how we formulate the problem statement is provided. We firstly introduce the research done in the field to provide some context to the reader, then we shift focus on the past works done to analyze the rumor spread and detection in Twitter. Some state-of-the-art overview in this research domain is:

1. *THEORIES OF RUMOR TRANSMISSION*

Allport and portman in their earlier work psychology of rumor” [3] stated that “rumors become more concise, easy to grasp and told as it propagates.” In this work, the rumor life cycle is divided into three phases namely leveling, sharpening, and assimilation. They gave a law of rumor that provides a basic relation between rumor transmission state and the importance of the topic. It is stated as:

Where R, i and a denote rumor spread intensity, importance of the topic, and ambiguity, respectively. It means that if the rumor topic is not important to a certain audience, then the spread comes to a halt in that group. Generally, the emergence of a rumor is to fulfill a particular function.

1. *EARLY WORKS*

Castillo et al. (2011 [4]) is one of the earliest works in developing statistical and automatic approach to detect rumors with the intention to learn and engineer a wide range of attributes from posts, user profiles and patterns of tweet propagation to develop a supervised classifier for detection. Later works by Frigger et al. (2014), Hannak et al. (2014) conducted studies to craft new features representing rumor diffusion. Kwon et al. (2013) used a time-series model on tweets with varying propagation time. Liu et al. (2015), K. Wu et al. (2015), Yang et al. (2012) introduced numerous sets of features from various perspectives. As the field of AI is emerging, a few researchers are striving to recognize rumors through deep learning, Ma et al. (2018) and Rath et al. (2017) made a model using Recurrent neural network of deep learning to understand the abstract expression of rumors. Guo et al. (2018) combined feature engineering and deep learning model to form a hierarchical social attention model that led to the improvement of rumor detection models. The most critical challenge of rumor detection is early detection of rumors, Wang et al. (2017) and Kwon et al. (2017) proposed a probabilistic model based on the eminent features of propagation of rumors. It was found that the user and linguistic features can be used as principal methods to indicate rumors. Along with the traditional models, deep learning and machine learning have been applied to detect rumors early. Wu et al. (2015) made a model to capture high-order propagation patterns using a graph-kernel based hybrid SVM classifier. Zhao et al. (2015) proposed a technique for searching enquiry phrases with high performance. Zhang et al. (2018) developed an early rumor detection heterogenous network which yielded 61% precision. T. Chen et al. (2018) made a soft-attention structure using RNN. L. Wu et al. (2017) made a neural network model. They deduced rumor categories, selected discriminative features, and learnt a rumor classifier. Nyugen et al. (2017) proposed a CNN for understanding the hidden representations of each tweet when combined with a time series.

IV. METHODS

In this section, we introduce the four hypotheses which we assumed to have a good and discriminative effect on the detection of rumor in tweets.

1. ***User tweet metadata***

In this hypothesis, we have used combined metadata of the user database and tweet database. We explored various features and tried extracting more discriminative features from the already existing ones in the database. (Discuss: Databases used are Twitter 15-16 and Pheme dataset). We further analyzed the extracted features for understanding the data and for performing feature selection.

1. ***Log transform (Method for converting skewed features into normal distributions)***

Incrementing all the non-categorical skewed features by one followed by performing log transformation on non-categorical skewed features, which converts the data into a normal distribution. Normal distribution ensures that more reliable predictions are made from the available data. The non-categorical skewed features were incremented by 1 to avoid the mathematical error of which is undefined.

1. ***Point biserial correlation (Feature selection method)***

This method is a special case of Pearson’s correlation. It is used to find the correlation between a continuous variable and a binary variable.

where,

= M[ean](https://www.statisticshowto.com/mean/) of the group that received a positive binary variable.

= Mean of the group that received a negative binary variable.

= Standard deviation

= Proportion of cases in the “0” group

= Proportion of cases in the “1” group

The resultant correlation coefficient ranges between 0 and 1.

If,

, then it signifies a positive correlation between the features.

, then it signifies a negative correlation between the features.

then it signifies no correlation between the features.

1. ***Mutual information correlation (used for feature selection)***

It is a measure of mutual dependence between two features, it uses the theory of entropy.

1. ***Text data of tweets***

In this hypothesis, we use text in tweets to create a model which supports the detection of rumors in tweets.

1. ***Word embeddings***

It is a representation of words in the form of a real-valued vector. It is encoded such that words with similar meanings get closer vectors in the vector space. We have used an open-source pre-trained word embedding called fastText [5].

1. ***Fake or real***

In this hypothesis, we use the text data to determine the probability of the data being fake or real. Example: - “If person X tells person Y that person Z is smart, this news is still a rumor to person Y. Later, if person A tells person Y that the news that person Z is smart is true, then the probability of being a non-rumor of the news that person X told to person Y about person Z increases.”

The methods used in the process of training this model are similar to that of the [Text data of the tweets](https://docs.google.com/document/u/0/d/1QFlvMXTtwVQ6KyCvDN4uSu6hgXQsncSx3Qyz2PTqoG4/edit) model.

1. ***Sentiment analysis on the reaction of tweets***

In this hypothesis, we use the reaction(comment) data linked with the tweets. Sentiment analysis has been done on the text data of the reactions linked to the tweets and polarity received from the analyzer is weighted by multiplying with metadata variable values of the reaction.

\*\* To be filled\*\*

Now, architecture, various hyperparameters relating to the models will be explained.

1. ***Architecture***
   1. ***Activation functions***

The activation function of the layer determines the output of the layer given a set of inputs. Activation functions used in the implemented neural network models are.

* Relu
* Sigmoid

Relu: It gives the output directly if the output is positive else gives 0.

where,

\*fill\*

The output ranges between

Sigmoid:

It is a real, bounded, and differentiable function defined for all input values. The output ranges between .

1. ***Weight initialization***

Weight initialization is a procedure to set the initial weights of a neural network using different methods. Weight initialization methods used in the implemented neural network models are.

* He-normal
* Glorot normal

He-normal: In this method of weight initialization, weights are initialized by considering the size of the previous layer. This helps in faster and accurate gradient descent.

It draws weights from a normal distribution centered on 0 and with a standard deviation as calculated from the formula below.

For relu,

where,

, is the output of layer before applying activation function.

,, are weights of respective nodes .

, is the number of inputs units.

He-normal is used with relu activation functions in the implemented neural networks.

Glorot-normal: In this method of weight initialization, we use fan-in and fan-out to determine the initial weights of the nodes. It draws weights from a normal distribution centred on 0 and with a standard deviation as calculated from the formula below.

where,

, is the number of inputs units.

, is the number of outputs units.

Glorot-normal is used with sigmoid activation functions in the implemented neural networks.

* 1. ***Batch Normalization***

This is a technique that transforms the data in a way that it has a mean of 0 and a standard deviation of 1. In cases where the input to the initial layer of the neural network is already normalized, the output from the activation functions might no longer be on the same scale and hence causing internal covariate shifts in the data. Batch normalization is applied to each input to avoid this problem.

Batch normalization is a two-step process. First, we normalize the input, and later rescaling and offsetting are performed.

We first compute the mean using the input from the i-th layer,

where,

, are the inputs from layer h

, is the number of neurons in layer h.

Next, we calculate the standard deviation of hidden activations.

Now, we subtract the mean from each input and divide by the sum of standard deviation and smoothing term to get the normalized data.

In the final operation, we use rescaling parameter and shifting parameter .

1. ***Loss functions***

Loss function is used to determine the error between the output and the desired output. The primary loss function used for classification problems is cross entropy. We categorize tweets into either rumour or non-rumour so, it has only 2 outputs. Hence, binary cross entropy is used.

where,

is the desired output.

, is the predicted output.

1. ***Adaptive moment estimation (Adam)***

It is a combination of gradient descent with momentum and RMSProp. It gets smoothening factor from momentum and adaptive learning rate like in RMSProp.

First, we calculate weighted average of past gradients,

where,

, the exponentially weighted average of past gradients.

, the exponentially weighted average of past gradients of bias.

, cost gradient with respect to current layer.

Next, we calculate the exponentially weighted average of the squares of past gradients.

, the exponentially weighted       average of past squares of gradients.

, the exponentially weighted average of past squares of gradients of bias.

Calculated averages have a bias towards zero so to fix this we use the technique of bias correction.

Now, the final formula for the weights will be calculated as follows.

1. EXPERIMENTS

In this section, the overall research is divided into 5 parts. The parts are divided based on the hypotheses. Each part will contain details regarding the database used, pre-processing, selected features, the model used.

***A. TWEET TEXT ANALYSIS MODEL***

***1) DATABASE***

In this hypothesis, the combination of PHEME, Twitter 15 and Twitter 16 is used.

***2) PRE\_PROCESSING***

Initially, we used the tweet-pre-processor [6] library of python to remove links, emojis, numbers. Next, we used the contractions [7] library of Python to replace the contracted words with their expanded forms. Next, we pre-compiled a list of abbreviation which are commonly used in tweets and then we replaced the abbreviated words in Tweet text with their true forms. Next, we used the word segment [8] library of Python to remove mentions, removing the hashtags and splitting the text in the hashtags. Next, we removed the stop words using a pre-defined list. Next, we used the Tokenizer [8] library of Python to fit on texts and then convert them to sequences. We further padded the size of the sequences to a size that is equal to the sum of the mean of cleaned tweets length and standard deviation of the length of cleaned tweets. We have used fast Text word embeddings to create the embedding matrix which is further passed into the neural network.

***3)TRAINING AND MODEL ARCHITECTURE***

We have used a sequential TensorFlow model. The first layer in the model is an Embedding layer which takes the embedding matrix, embeddings dimension, input length as inputs. Next, we have used a dropout layer with a value of 0.5. Next, we have used 4 layers of Bidirectional LSTM with 64 nodes in each with return sequences turned on. We further have 1 layer of Bidirectional LSTM with 32 nodes. Then, we have a dense layer with 32 nodes and a relu activation function. Next, we have a layer of Batch Normalization which is processed with a dropout layer with a value of 0.5. Further, we have the final layer which is a dense layer with 1 node with the activation function of the sigmoid. This model is compiled with Adam optimizer and the loss used is binary cross-entropy. The accuracy metrics used is accuracy. This model summary is as follows.

|  |  |  |
| --- | --- | --- |
| **Layer (Type)** | **Output Shape** | **Param#** |
| embedding (Embedding) | (None,93,300) | 2724600 |
| dropout (Dropout) | (None,93,300) | 0 |
| bidirectional (Bidirectional LSTM) | (None,93,128) | 186880 |
| Bidirectional\_1(Bidirectional LSTM) | (None,93,128) | 98816 |
| bidirectional\_2(Bidirectional LSTM) | (None,93,128) | 98816 |
| bidirectional\_3(Bidirectional LSTM) | (None,93,128) | 98816 |
| bidirectional\_4(Bidirectional LSTM) | (None,64) | 41216 |
| dense (Dense) | (None,32) | 2080 |
| batch normalization (Batch Normalization) | (None,32) | 128 |
| dropout\_1(Dropout) | (None,32) | 0 |
| dense\_1(Dense) | (None,1) | 33 |
| Total Params: 6,317,985 | | |
| Trainable params: 526,721 | | |
| Non-trainable params: 5,791,264 | | |

This data was split into test and train with a test size of 0.2, random state of 42 and stratified for the target labels.

The model was trained with the X\_train padded sequences and y\_train and with validation data ( X\_test padded sequences, y\_test). The batch size taken is 128 and trained for 150 epochs with an early stopping monitoring for minimum validation loss with patience of 5.

***B. FAKE TWEET CLASSIFICATION***

***1) DATABASE***

In this hypothesis, GossipCon dataset is used.

***2)PRE\_PROCESSING***

The pre-processing is the same as the Tweet text analysis model.

***3)TRAINING AND MODEL ARCHITECTURE***

The model used here is the same as the tweet text analysis model. Only the number of word embeddings is changed due to different sizes of databases. This model summary is as follows.

|  |  |  |
| --- | --- | --- |
| **Layer (Type)** | **Output Shape** | **Param#** |
| embedding (Embedding) | (None,84,300) | 5791200 |
| dropout (Dropout) | (None,84,300) | 0 |
| bidirectional (Bidirectional LSTM) | (None,84,128) | 186880 |
| Bidirectional\_1(Bidirectional LSTM) | (None,84,128) | 98816 |
| bidirectional\_2(Bidirectional LSTM) | (None,84,128) | 98816 |
| bidirectional\_3(Bidirectional LSTM) | (None,84,128) | 98816 |
| bidirectional\_4(Bidirectional LSTM) | (None,64) | 41216 |
| dense (Dense) | (None,32) | 2080 |
| batch normalization (Batch Normalization) | (None,32) | 128 |
| dropout\_1(Dropout) | (None,32) | 0 |
| dense\_1(Dense) | (None,1) | 33 |
| Total Params: 6,317,985 | | |
| Trainable params: 526,721 | | |
| Non-trainable params: 5,791,264 | | |

This data was split into test and train with a test size of 0.2, random state of 42 and stratified for the target labels.

The model was trained with the X\_train padded sequences and y\_train and with validation data ( X\_test padded sequences, y\_test). The batch size taken is 256 and trained for 150 epochs with an early stopping monitoring for minimum validation loss with patience of 5.

***C. USER AND TWEET META-DATA MODEL***

***1) DATABASE***

In this hypothesis, the combination of PHEME, Twitter 15 and Twitter 16 is used

***2)PRE\_PROCESSING***

The data was analysed and the features with skewed data were log-transformed to get a normal distribution. Next, the pre-processing of text is done like the tweet text analysis model. Further, the text was also lemmatized. We used TextBlob [9] to find the sentiment polarity of the text. Next, we calculated the text length and tweet-age features. Next, we calculated the Posted in feature which is the time gap in which the user created the profile and posted the tweet. The final feature list is.

|  |  |
| --- | --- |
| **Numerical Features** | **Categorical Features** |
| Follower’s count | Is\_reply |
| Retweet count | Verified |
| Favourite count | Is\_quote\_status |
| No: of Symbols | Profile\_image\_url |
| No: of User mentions | profile\_background\_image\_url |
| No: of Hashtags | Default profile image |
| No: of URL’s | Default profile |
| Polarity | Profile\_use\_background\_image |
| Text length | Has\_location |
| Post age | Has\_url |
| Statuses count |  |
| Friends count |
| Favourites count of user |
| Listed count |
| Account age |
| Screen name length |
| The time gap between user-created time and tweeted time(Posted\_in\_time) |

In total, we have 27 features. We performed feature selection on these features and arrived at the final features.

The final list of selected features after feature selection is.

|  |  |
| --- | --- |
| **Numerical Features** | **Categorical Features** |
| Text Length | Profile\_use\_background\_image |
| No: of URL’s | Profile\_background\_image\_url |
| Post age | Default\_profile\_image |
| Statuses count | Default\_profile |
| Listed count | Verified |
| No: of symbols |  |
| Posted\_in |
| No: of user mentions |
| Polarity |
| Favourites count of user |
| Favourite count of a tweet |
| Screen name length |
| No: of hashtags |

***3)TRAINING AND MODEL ARCHITECTURE***

First, we applied a Standard scaler to the selected features. We have used a sequential TensorFlow model. The first layer is a dense layer with 64 nodes, relu activation function and L2 regularization with a value of 0.01. Next, we have a dropout layer with a value of 0.3. Further, we have a dense layer with 64 nodes, relu activation function and L2 regularization with a value of 0.01. Next, we have a dropout layer with a value of 0.3. The final layer is a dense layer with a sigmoid activation function and glorot\_uniform weights initializer. The model is compiled with Adam optimizer, having binary cross-entropy loss and metrics as accuracy. This model summary is as follows.

|  |  |  |
| --- | --- | --- |
| **Layer(type)** | **Output Shape** | **Param #** |
| dense\_1(Dense) | (None,64) | 1216 |
| dropout\_1(Dropout) | (None,64) | 0 |
| dense\_2(Dense) | (None,64) | 4160 |
| dropout\_2(Dropout) | (None,64) | 0 |
| dense\_3(Dense) | (None,1) | 65 |
| Total params: 5,441 | | |
| Trainable params: 5,441 | | |
| Non-trainable params: 0 | | |

This data was split into test and train with a test size of 0.2, random state of 42 and stratified for the target labels.

The model was trained with scaled X\_train values and y\_train. The batch size taken was 32 and validation data was (X\_test\_scaled,y\_test).

***D. SENTIMENTAL ANALYSIS ON REACTION OF TWEETS MODEL***

***1) DATABASE***

In this hypothesis, the PHEME dataset is used.

***2)PRE\_PROCESSING***

The text comments on the tweets are cleaned and pre-processed like the tweet text analysis model and further, it is lemmatized. Text Blob Python library was used to find the polarity of the comments on the tweets. We multiplied these polarities with favourite count, retweet count, sum of favourite count and retweet count to get 3 weighted averages of polarities. Retweet count, Favourite count, no: of hashtags in the text of comments are also taken in the total features list. The total list of features is.

|  |
| --- |
| Retweet Count |
| Favourite count |
| Sentiments |
| No: of hashtags |
| Favourite weighted polarity |
| Retweet weighted polarity |
| Favourite retweet weighted polarity |

After performing feature selection, it was observed that Favourite weighted polarity, retweet weighted polarity and favourite retweet weighted polarity had a similar negative correlation to the label.

So, we only took the main polarity(sentiments) feature during feature selection.

The final list of selected features is.

|  |
| --- |
| Favourite count |
| Sentiments (Polarity) |
| No: of hashtags |

This data is scaled using Standard Scaler after splitting it into test and train.

***3)TRAINING AND MODEL ARCHITECTURE***

We have used a sequential TensorFlow model. The first layer is a dense layer with 32 nodes and a relu activation function. Second, we have a dropout layer with a value of 0.3. Third, we have a dense layer with 16 nodes and a relu activation function. Fourth, we have a dropout layer with a value of 0.2. Fifth, we have a dense layer with 16 nodes and a relu activation function. Sixth, we have the final layer which is a dense layer with a sigmoid activation function.

This model is compiled with Adam optimizer and binary cross-entropy is taken as loss function. The metric used is accuracy. The model summary is as follows.

|  |  |  |
| --- | --- | --- |
| **Layer(type)** | **Output Shape** | **Param #** |
| dense\_1(Dense) | (4598,32) | 256 |
| drouput\_1(Dropout) | (4598,32) | 0 |
| dense\_2(Dense) | (4598,16) | 528 |
| dropout\_2(Dropout) | (4598,16) | 0 |
| dense\_3(Dense) | (4598,16) | 272 |
| dense\_4(Dense) | (4598,1) | 17 |
| Total params: 1,073 | | |
| Trainable params: 1,073 | | |
| Non-trainable params: 0 | | |

This data was split into test and train with a test size of 0.2, random state of 42 and stratified for the target labels.

The model was trained with scaled X\_train values and y\_train. The batch size taken was 32 and validation data was (X\_test\_scaled,y\_test).

***E. ENSEMBLE MODEL COMBINING THE ABOVE MODELS***

***1) DATABASE***

In this hypothesis, the combination of PHEME, Twitter 15 and Twitter 16 is used

***2)TRAINING AND MODEL ARCHITECTURE***

Due to the lack of data of reactions on tweets, we have ensembled the other 3 models on the complete data and ensembled all 4 models only for PHEME data.

We have extracted predict probabilities of each model and treated them as separate features to train a model. We have used a sequential TensorFlow model. The first layer is a dense layer with 64 nodes and a relu activation function. The second is a dropout layer with a value of 0.3. Third, is a Dense layer with 64 nodes and a relu activation function. Fourth, is a dropout layer with a value of 0.3. The fifth and final layer is a dense layer with 1 node, sigmoid activation function and glorot\_uniform weight initializer.

The model is compiled using Adam optimizer and binary cross-entropy as loss function. The metrics used are accuracy. The summary of the model is as follows.

|  |  |  |
| --- | --- | --- |
| Layer (Type) | Output Shape | Param # |
| dense\_1(Dense) | (None,64) | 256 |
| droupout\_1(Dropout) | (None,64) | 0 |
| dense\_2(Dense) | (None,64) | 4160 |
| dropout\_2(Dropout) | (None,64) | 0 |
| dense\_8(Dense) | (None,1) | 65 |
| Total params: 4,481 | | |
| Trainable params: 4,481 | | |
| Non- trainable params: 0 | | |

1. RESULTS

The results from each of the models discussed in the experiments section are shown tabularly in this section.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Loss** | | **Accuracy (in %)** | |
| **Training** | **Validation** | **Training** | **Validation** |
| Tweet text analysis model | 0.274 | 0.449 | 88.6 | 80.1 |
| Fake tweet classification model | 0.074 | 0.084 | 97.3 | 97.4 |
| User and tweet meta data model | 0.469 | 0.550 | 78.1 | 71.9 |
| Sentiment analysis on reaction of tweets | 0.644 | 0.647 | 63.5 | 63.3 |

Below is the result from the ensembled model of Tweet text analysis model, Fake tweet classification model, User and tweet metadata model on the complete dataset (PHEME + Twitter 15 + Twitter 16).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Loss** | | **Accuracy (in %)** | |
| **Training** | **Validation** | **Training** | **Validation** |
| Ensemble model | 0.136 | 0.495 | 94.9 | 81.9 |

1. CONCLUSION, CHALLENGES AND FUTURE WORK

The features selected have a good correlation with the label and are useful to classify a tweet into a rumour or a non-rumour. But due to the limited availability of data, this cannot be concretely proved. Twitter deletes data from its servers on a timely basis which makes it impossible to get data as the databases used gets older.

Future work to make the models better include:

* Live training of ensembled models on live data to improve the accuracy of the model.
* Better analysis of comments on tweets by classifying them into supportive, rejective, question comments. This might do a better job in helping detect rumours in the primary tweet.
* Finding more databases to concretely prove our hypotheses.

APPENDIX

ACKNOWLEDGMENT

REFERENCES

[1] Peterson, W., & Noel, G. (1951, September). Rumor and Punlic Opinion. *The American Journal of Sociology*, 159-167. doi:10.1086/220916

[2] Mahudeswaran, D., & Shu, K. (n.d.). *FakeNEwsNet*. Retrieved from Github.com: <https://github.com/KaiDMML/FakeNewsNet>

[3] W., A., & Leo, J. P. (1945, December). The Basic Psychology of Rumor. *Transactions of the New York Academy of Sciences*, 61-81. doi:10.1111/j.2164-0947.1945.tb00216.x

[4] Castillo, C., Mendoza, M., & Barbara, P. (2013, October). Predicting information credibilityin time-sensitive social media. *Internet Research*, 560-588. doi:10.1108/IntR-05-2012-0095

[5] Grave, E., Joulin, A., Bojanowski, P., & T., M. (2016). Bag of Tricks for Efficient Text Classification. *arXiv preprint*.

[6] Özcan, S. (n.d.). *Tweet-preprocessor*. Retrieved from pypi.org: <https://pypi.org/project/tweet-preprocessor/>

[7] Jenks, G. (n.d.). *Python Word Segmentation*. Retrieved from grantjenkis.com: <http://www.grantjenks.com/docs/wordsegment/>

[8] Google. (n.d.). *Tensorlfow Core v2.5.0*. Retrieved from tensorflow.org: https://www.tensorflow.org/api\_docs/python/tf/keras/preprocessing/text/Tokenizer

[9] Keen, P., Honnibal, M., & Yankovsky, R. (n.d.). *TextBlob: Simplified Text Processing*. Retrieved from https://textblob.readthedocs.io: https://textblob.readthedocs.io/en/dev/

FIRST A. AUTHOR (M’76–SM’81–F’87) and all authors may include biographies. Biographies are often not included in conference-related papers. This author became a Member (M) of IEEE in 1976, a Senior Member (SM) in 1981, and a Fellow (F) in 1987. The first paragraph may contain a place and/or date of birth (list place, then date). Next, the author’s educational background is listed. The degrees should be listed with type of degree in what field, which institution, city, state, and country, and year the degree was earned. The author’s major field of study should be lower-cased.

The second paragraph uses the pronoun of the person (he or she) and not the author’s last name. It lists military and work experience, including summer and fellowship jobs. Job titles are capitalized. The current job must have a location; previous positions may be listed without one. Information concerning previous publications may be included. Try not to list more than three books or published articles. The format for listing publishers of a book within the biography is: title of book (publisher name, year) similar to a reference. Current and previous research interests end the paragraph.

The third paragraph begins with the author’s title and last name (e.g., Dr. Smith, Prof. Jones, Mr. Kajor, Ms. Hunter). List any memberships in professional societies other than the IEEE. Finally, list any awards and work for IEEE committees and publications. If a photograph is provided, it should be of good quality, and professional-looking. Following are two examples of an author’s biography.

**SECOND B. AUTHOR** was born in Greenwich Village, New York, NY, USA in 1977. He received the B.S. and M.S. degrees in aerospace engineering from the University of Virginia, Charlottesville, in 2001 and the Ph.D. degree in mechanical engineering from Drexel University, Philadelphia, PA, in 2008.

From 2001 to 2004, he was a Research Assistant with the Princeton Plasma Physics Laboratory. Since 2009, he has been an Assistant Professor with the Mechanical Engineering Department, Texas A&M University, College Station. He is the author of three books, more than 150 articles, and more than 70 inventions. His research interests include high-pressure and high-density nonthermal plasma discharge processes and applications, microscale plasma discharges, discharges in liquids, spectroscopic diagnostics, plasma propulsion, and innovation plasma applications. He is an Associate Editor of the journal *Earth*, *Moon*, *Planets*, and holds two patents.

Dr. Author was a recipient of the International Association of Geomagnetism and Aeronomy Young Scientist Award for Excellence in 2008, and the IEEE Electromagnetic Compatibility Society Best Symposium Paper Award in 2011.

**THIRD C. AUTHOR, JR.** (M’87) received the B.S. degree in mechanical engineering from National Chung Cheng University, Chiayi, Taiwan, in 2004 and the M.S. degree in mechanical engineering from National Tsing Hua University, Hsinchu, Taiwan, in 2006. He is currently pursuing the Ph.D. degree in mechanical engineering at Texas A&M University, College Station, TX, USA.

From 2008 to 2009, he was a Research Assistant with the Institute of Physics, Academia Sinica, Tapei, Taiwan. His research interest includes the development of surface processing and biological/medical treatment techniques using nonthermal atmospheric pressure plasmas, fundamental study of plasma sources, and fabrication of micro- or nanostructured surfaces.

Mr. Author’s awards and honors include the Frew Fellowship (Australian Academy of Science), the I. I. Rabi Prize (APS), the European Frequency and Time Forum Award, the Carl Zeiss Research Award, the William F. Meggers Award and the Adolph Lomb Medal (OSA).