Debunking Fake News by Leveraging Speaker Credibility and BERT Based Model

Thoudam Doren Singh
Department of Computer Science and Engineering
National Institute of Technology, Silchar
Silchar, India
doren@cse.nits.ac.in

Divyansha
Department of Computer Science and Engineering
National Institute of Technology, Silchar
Silchar, India
divyansha_ug@cse.nits.ac.in

Apoorva Vikram Singh
Department of Electrical Engineering
National Institute of Technology, Silchar
Silchar, India
apoorva.vsingh@ieee.org

Anubhav Sachan
Department of Electronics and Communication Engineering
National Institute of Technology, Silchar
Silchar, India
anubhav_ug@ece.nits.ac.in

Abdullah Faiz Ur Rahman Khilji
Department of Computer Science and Engineering
National Institute of Technology, Silchar
Silchar, India
abdullah_ug@cse.nits.ac.in

***Abstract*—The exponential growth in fake news and its role in deteriorating general public trust and democratic standards certainly calls for some counter combat approaches. The prediction of chances of news to be fake is deemed to be hard task since most of the deceptive news has its roots in true news. With a minor fabrication in legitimate news, influential fake news can be created that can be used for political, entertainment, or business-related gains. This work provides a novel intuitive approach to exploit data from multiple sources to segregate news into real and fake. To efficiently capture the contextual information present in the data, Bidirectional Encoder Representations from Transformer (BERT) have been deployed. It attempts to further enhance the performance of the deceptive news detection model by incorporating information about the speaker profile and the credibility associated with him/her. A hybrid sequence encoding model has been proposed to harvest the speaker profile and speaker credibility data which makes it useful for prediction. On evaluation over benchmark fake news dataset LIAR, our model outperformed the previous state-of-the-art works. This attests to the fact that the speaker's profile and credibility play a crucial role in predicting the validity of news.**

***Keywords*-Fake News, Text Classification, BERT, LIAR**

I. Introduction

With the substantial growth in the means of propagation of news and agendas, the boundary between news reporting and news creation seems to be fading. Fake news can be identified as one of the most significant threats to journalism and hence democracy. Content producers (mostly journalists) are galvanized towards paid news and spectacle over precise content [1]. The consumers of this content usually lack the cognition to scrutinize news critically. This leads to the deterioration of public trust in journalism and government. The effects of fake news propagation can be vividly observed in incidents like the 2016 U.S. presidential elections where it allegedly proved to be decisive in the results [2].

This subtle fiasco of news agencies calls for some automated verifying tool that can flag and filter false news. The commonly pursued approach to deal with this problem is called fact-checking [3]. Fact-checking uses the collective intelligence of a large group of regular individuals. However, keeping up the manual fact-checking processes with the huge scale of the data produced daily is an exhausting business. Under these circumstances, automated fact-checking methods have gained ground. The automated fact-checking methods hugely rely on Natural Language Processing (NLP) methods, Information Retrieval (IR) techniques, and Graph Theory [4].

The methods devised to combat fake news are persistently met with some challenges that make it hard to design a system that is accurate and efficient at the same time. Most of the fake news circulated is not absolutely fake. It is originated by tweaking a real news story for entertainment, for pure malice, or political advantages. Due to this, linguistic and factual proximity of fake news with actual news is diminished. Higher this proximity is, harder it is to segregate them into real and fake. The low availability of labeled fake and true real-world data also hampers the process of the development of fake news detection techniques.

In this work, we attempt to incorporate the speaker's

profile and the speaker's credibility into the process of fake news detection. The speaker information used to check the legitimacy of the statement comprised of the profession, political party affiliation, state of residency, subject on which statement is made and venue at which statement is made. Speaker's credibility has been calculated by probabilistic estimation of the frequency of release of fake statements by the speaker. It has then been used as an attention factor. Our results on a benchmark fake news dataset on evaluation proved to be an improvement over previous approaches on the dataset.

The rest of the paper is organized as the following. In section II, we discuss the previous works done in the field of fake news classification. The section III details the various datasets that have been used for the training and validation of the model. The section IV gives a synopsis of the complete idea behind the work. Following it, the section V contains the technique used later in the paper to refocus the features. The section VI highlights the different types of feature extraction techniques used for different types of the data utilized. The section VII explains the architecture used to exploit the different types of features generated in the previous section. The section VIII contains the experimental details of the proposed model. Finally, we discuss the results and give some future prospects of the work in section IX.

II. Related works

Detection of fake news has been an active area of research in past few years due to the high risks associated with it. A substantial amount of work has been done in studying fake news and misinformation detection. One well known technique is to take advantage of the linguistic properties of the news article. Fake news contains unusual language [5]. Features like POS (Part of Speech), n-gram and context free grammar based production rules are used for deception detection [6]. Despite the success of such methods there is no optimized set of features that can be used. Many researchers use neural network based approach owing to its ability of automatic feature extraction. On the basis of content of the post and interactions of user at different times, [7] used recurrent neural network (RNN) for rumour detection and outperformed models that used hand crafted features. However these techniques fail to perform on newly emerged news [6] since they tend to learn event-specific features that can not be transferred to unseen events.

Recently, hybrid models have become popular in this domain as they use linguistic properties as well as some extra information about the news like speaker's profile. LIAR dataset [8] was introduced to facilitate building such models. LIAR consists of speaker's information and their credit history along with the lexical features. There are many ways to incorporate this extra information into the hybrid model. To determine the credibility of news based on speaker's reputation, speaker profile can be used [9]. To propose a hybrid long short-term memory (LSTM) model, [10] used speaker profiles as an attention factor. Long term user credibility relations have a role to play in detecting Fake News. Different users have different credibility levels on social media [11]. Generally, users having lesser credibility score imply having a greater chance of spreading fake news. The credibility features can be effectively used to further improve the performance [12].

In this work, we conduct a multi-class classification of news to determine how much true or fake a piece of news is based on the classification labels used.

III. Dataset

A. LIAR

We evaluate our results on a benchmark fake news detection dataset LIAR [8]. The choice of a political news dataset was due to the fact that false political news travels nearly three times faster and appeals to a larger audience than news of any other category [13]. It also has a deeper diffusion among the masses reaching greater depths than other categories of news.

LIAR consists of 12,836 short labeled statements from 3,341 different speakers. It covers 141 different topics from the fact-checking website politifact[1]. Every statement from LIAR consists of speaker profile, topic and statement by the speaker. Speaker profile includes speaker name, the home state of the speaker, credit history, political party affiliation, current job, title and venue of the speech. 18 tokens are present on an average per statement. Six different labels are present stating the degree of legitimacy of the statement: true, half-true, mostly-true, barely-true, false and pants-fire. The dataset has been split into 90% for training and 10% for validation. The figures of LIAR are described in Table I with an example of statements made by Mr. Barack Obama.

B. Speaker2Credit

Speaker2Credit [12] dataset contains the credit history for each speaker. Credit history maintains the count of inaccurate statements made by the speaker in the past. For example, Mr. Donald Trump has a credit history c = (true = 6, mostly-true = 27, half-true = 39, barely-true = 48, false = 102, pants-fire = 52).

The intuition behind using the speaker's credibility for detecting fake news is that if one has an idea about the speaker's past records while making a statement, one can estimate the speaker's tendency to tell truth or lie.

IV. A Brief Overview

In recent years, it has been noticed that a single predictive model fed with an amalgamation of different forms of information has the potential of improving the performance of any machine learning technique. With this motivation,

[1]https://www.politifact.com/

Table I
DATASET COUNT PER RULING

Ruling	Count of Ruling	Sample Statement
true	2,063	McCain opposed a requirement that the government buy American-made motorcycles. And he said all buy-American provisions were quote 'disgraceful'.
mostly-true	2,466	The cost of health care is now the single-biggest factor driving down the federal budget deficit.
half-true	2,638	I'm the only person on this stage who has worked actively just last year passing, along with Russ Feingold, some of the toughest ethics reform since Watergate.
barely-true	2,108	I introduced a bill that wouldve helped stop the multimillion-dollar bonus packages that CEOs grab on their way out.. (McCain) opposed that idea.
false	2,511	Twelve judges have thrown out legal challenges to the health care law because they rejected the notion that the health care law was unconstitutional.
pants-fire	1,050	What we said was, you can keep (your plan) if it hasnt changed since the law passed.

we leverage three different types of data as inputs to our proposed fake news detection model. Firstly, we use lexical features extracted from the actual statements of the speakers. In addition to this, the speaker's profile as the metadata for the model has been used. Speaker's profile can have a significant impact on the decision-making process for any news to be fake or real. For example, a conservative might undermine notions like abortion rights while a progressive might exaggerate ideas like the removal of internet censorship. Lastly, credit history has been used from the Speaker2Credit dataset to inculcate the credibility of the speaker in the decision-making process to infer news as fake.

We introduce a hybrid model that treats fake news detection as a multi-class classification problem. A sequential model is utilized to encode the speaker's statements. Speaker profile information is then added and the speaker's credibility is used as an attention [14] factor to form a hybrid model.

The different architectures used for sequential encoding of the speaker's statements include Long short-term memory (LSTM) networks [15], Convolutional Neural Network (CNN) [16], LSTM-CNN [17], Recurrent-CNN (RCNN) [18], etc. as discussed in section VIII-A.

The one-hot encoded vectors of the speaker's profile and credit vectors have been incorporated in the model by the means of simple concatenation to the encodings of the statements obtained from the sequential model. We also use a simple attention mechanism to refocus the statement encodings in accordance with the speaker's credibility as discussed in section V.

V. REFOCUSING

The Mathematical Tensor Product [19] proposes a way in which the procedural information of tensor product is combined with the encoding of different states included in its vector representation. In quantum mechanics if the unit vector J of a quantum be represented by r observations in any given time frame, the vector can be postulated to have r dimensions belonging to hilbert Space C^r. The interactions of the two similar quanta is represented be a density matrix $C^n \times C^m$ accounting for all $n \times m$ interactions. Since the tensor product captures the way things interact, it can be extended to accommodate all possible interactions between the encoded sentences and the credibility vectors. The resultant thus would have the required information necessary for classification tasks.

We use the Customized Tensor Product (CTP) to refocus the sentence encoding ($\overrightarrow{A}$) obtained from the sequence encoder as discussed later in section VII. The speaker's credit history is taken as $\overrightarrow{B}$. Let n_a and n_b be the number of elements in A and B respectively. The actual tensor product gives a vector of length $n_a \times n_b$. Thus, giving a new space wherein the dimensions multiply. The mathematical tensor product obtained has the properties of both matrix as well as vector operations. For our work, $\overrightarrow{B} \otimes \overrightarrow{A}$ has been used wherein we focus the encoded text onto the credibility vectors thus obtaining rich features suitable for training; we call this as $\overrightarrow{C}$ (as given by equation 1).

$$\overrightarrow{C} = \overrightarrow{B} \otimes \overrightarrow{A} \tag{1}$$

On the resultant $\overrightarrow{C}$ obtained we employ the use of 1D-convolution operation with stride and kernel size equivalent to the size of $\overrightarrow{B}$ to obtain the resultant $\overrightarrow{Z}$ which is a n_a dimensional vector. Here Z is the final output of our CTP. For suitable optimisation and fast computation we employ the tensor operations to get the resultant $\overrightarrow{Z}$ of size n_a (given by equation 2).

$$Z_i = \sum_{a=0}^{n_{b-1}} D_a \times C(i \cdot n_b + a), \forall i \in [0, n_a) \tag{2}$$

Here, $\overrightarrow{D}$ is an all ones vector of size n_b (refer to equation 3).

$$\overrightarrow{D} = 1_{n_b} \tag{3}$$

Z_i is the i^{th} element of $\overrightarrow{Z}$. Thus, the CTP function can be given by the equation 4.

$$CTP(\overrightarrow{A}, \overrightarrow{B}) = \overrightarrow{Z} \tag{4}$$

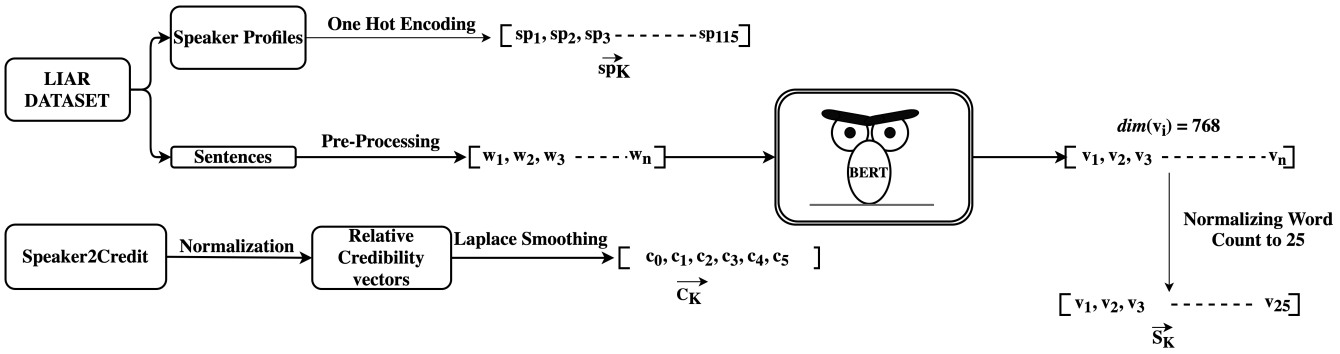

Figure 1. Feature Generation

VI. Feature Extraction

As discussed before, this work makes use of three different types of features. They include lexical features from the statements, speaker profile based meta-features and features based on the credibility of speaker deduced from the past. In this section, we discuss the approaches we have used to curate these features from the data and integrate them. An overview of feature extraction process is as shown in Fig. 1.

A. BERT for Statements Representation

Prior to generating lexical features from sentence embedding, we pre-process the statement data from the LIAR dataset. For this purpose, operations for punctuation removal, conversion of all characters to lowercase, tokenization, and stemming has been conducted on the statements. The length of the statements are fixed to 25 words based on the average length. The reason for this is to avoid excessive padding in short statements to comply with the BERT architecture [20].

We, then, use BERT model to create token embeddings for the sentences. BERT's rigorous training over a huge text corpus makes it better than most of the existing neural language models. As we train the model on the corpus (speaker), it stacks up an intimate and deeper understanding of the language. The bidirectionality of BERT ensures that it is absorbing information from both the right and left side of the token while training. It also solves the problem of the polysemy (different meaning of the same words depending on the context). For example, for a sentence- "he had a cell phone inside the prison cell", BERT will generate different embeddings for co-occurrences of "cell" in different contexts. This makes it superior for application over the speaker's statements. Solving an NLP problem using BERT is basically a two-step process. Initially, the language model is trained (semi-supervised or unsupervised) on a sizeable unlabeled text corpus. Finally, to harvest the knowledge this model has learned, the model is fine-tuned to perform specific NLP tasks (supervised).

However, instead of using this conventionally practiced approach, the process has been modified to suit our purpose. Following the first step, we obtain a pre-trained language model trained on wiki-corpus. However, for the next step we simply extract the embeddings from the BERT for the tokens in our statements and used them in our predictive model. We refrain from using the end-to-end classification models associated with BERT. This allows us to enhance the quality of the features used by integrating additional features with it.

The BERT model used for this purpose is BERT-Base uncased having 12 layers (transformer blocks), 768 hidden states, 12 attention heads and comprises 110 million parameters.

B. Speaker Profile Metadata

To further enrich the features obtained from the statement embeddings, we exploit the speaker's profile data supplied by the LIAR dataset. The metadata, as discussed before, contains the following attributes: job of the speaker, political party affiliation of the speaker, state of residency of the speaker, topic on which statement is made and the location of the speech. These factors can profoundly influence the actuality of the statements made by a speaker and hence need to be integrated with the features obtained from BERT.

The metadata used has been formatted in a manner to represent the diverse information in a lower dimension without losing out any vital information. The meta tags have been quantified into discrete classes and a representative 'rest' class that helped us to condense the classes. The discrete classes for top 20 speakers with highest number of speeches have been used. Only top 10 most common jobs have been allotted a discrete class. Similarly, top 5 parties, top 12 subjects and top 10 venues based on the value counts of their frequency of occurrence have been used. Furthermore, to cluster similar identifiers (for example to cluster 'Television interviews' and 'TV interviews' into similar category), substring matching has been used instead of using exact values from the dataset.

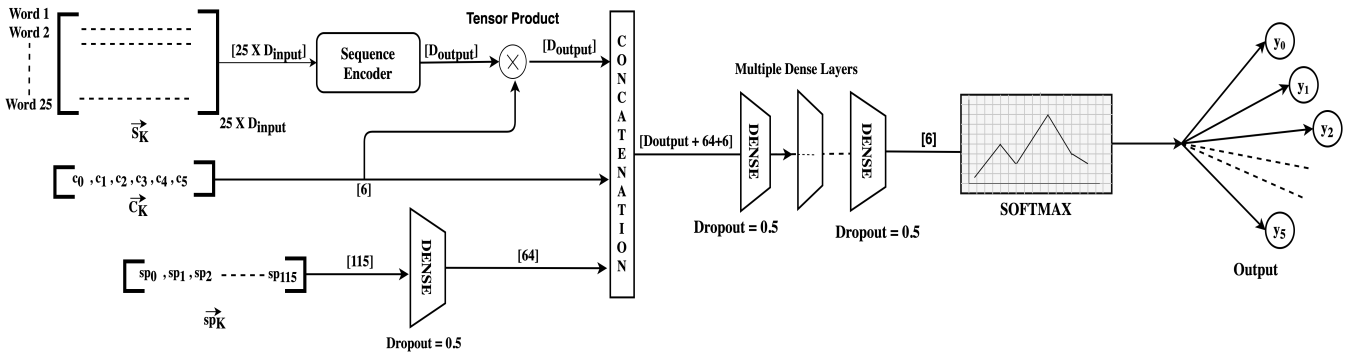

Figure 2. Proposed Architecture (The Number of Dense Layers in the Model depends on the corresponding value of D_{output})

To construct features from metadata, all the attributes of the speaker's profile have been converted into one-hot encoded representations and then concatenated to form a single vector of dimension 115.

C. Speaker Credibility Features

The drawback of using the credit history of LIAR as a metric of the speaker's credibility is that it is generated using only the information present in LIAR dataset itself and lacks the inclusion of the universal history of the speaker. This results in the generation of bias for the test set as it already contains information regarding the number of statements by every speaker under each label (true, half-true, etc.). Speaker2Credit [12] contains credit vectors for speakers taking into consideration their universal history of telling the truth or lie (irrespective of whether it is present in LIAR or not). Additionally, it contains the "true" label that was not present in the LIAR dataset. The universal history has been obtained by crawling through statements on politifact.com. Speaker2Credit consists of credit vectors of a total of 1,065 different speakers.

The credit vectors are responsible for identifying tendency of a speaker to lie rather than directly identifying a lie. We use the credit vectors present in the Speaker2Credit dataset with our own refinements to impart the information regarding the credibility of speakers into the prediction model. Further, we also calculate credit vectors for the speakers for whom no credit vectors were present in Speaker2Credit. This was done in order to maintain the consistency of the dataset. The data used for generating credit vectors for speakers whose credit vectors were absent from Speaker2Credit has been obtained from the LIAR dataset.

In order to refine the credit vectors for every speaker, we conduct simple mathematical operations on credit vectors of Speaker2Credit. Given a set of rulings[2] $r = \{\mu_0, \mu_1, \mu_2, \mu_3, \mu_4, \mu_5\}$ for a speaker i, its credit vector is given by $(count(\mu_0|i), count(\mu_1|i), \ldots, count(\mu_5|i))$. The relative credibility of the speaker is estimated by dividing the credit vector by the sum of all rulings in that vector. In order to avoid any null rulings, we utilize Laplace smoothing [21] or Lidstone smoothing. This technique is used to smooth categorical data. The smoothed relative credit vector is given by Equation 5 where $\alpha > 0$ is a smoothing parameter. According to Laplace's rule of succession [22], the value of α ideally should be 1.

$$credit_{i_\alpha} = \left(\frac{count(\mu_0|i) + \alpha}{\sum_\mu count(i) + 5\alpha} \cdots \frac{count(\mu_5|i) + \alpha}{\sum_\mu count(i) + 5\alpha} \right) \tag{5}$$

For unknown speakers, we use credit vectors from LIAR dataset and conducted the mathematical processing of the features as discussed above.

VII. Proposed Architecture

Once the task of feature curating is established, the acquired features are fed into our hybrid classification model. The model contains a sequence encoding block, an architecture to refocus the encoded features and finally a dense block to incorporate metadata into the refocused encoded features. The complete model can be seen in Fig. 2

Assuming N to be a compilation of news, SP to be the set of the speaker profiles and C to be the credit history of elements in set SP. A piece of news n_k, $n_k \in N$, comprises the tokens of text in it in the form of a sentence s_k, the profile of the speaker $\overrightarrow{sp_k} \in SP$ and his/her credit history vector represented by $\overrightarrow{c_k} \in C$. A sequence of words $w_{k1}, w_{k2}, w_{k3}, w_{k4} \ldots w_{kl}$ forms a sentence s_k having a length l. Each word w_i in s_k has a feature set forming a $wordvector = [F_{r_{1wki}}, F_{r_{2wki}} \ldots F_{r_{Dwki}}]$ of dimension D_{input}. A sentence vector $\overrightarrow{s_k}$ has been obtained as $[\overrightarrow{v}_{w_1}, \overrightarrow{v}_{w_2}, \overrightarrow{v}_{w_3} \ldots \overrightarrow{v}_{w_l}]$ having a dimension of $l \times D$. We use BERT embeddings to obtain word vectors as described in section VI-A. The profile vector $\overrightarrow{sp_k}$ is obtained by using one-hot encoding as described in section VI-B. Finally, the Credit vector $\overrightarrow{c_k}$ is obtained from the Speaker2Credit dataset as described section VI-C.

[2] μ_0 corresponds to "True" and $\mu_{d=5}$ to "Pants-on-Fire"

Table II
SAMPLE OUTPUT BASED ON PROPOSED MODEL PREDICTIONS WITH LSTM-CNN ENCODER.

Sr. No.	Statement	Prediction	Ruling	Match
1	Says Tennessee is providing millions of dollars to virtual school company for results at the bottom of the bottom.	True	True	Exact Match
2	State revenue projections have missed the mark month after month.	Half-True	Half-True	Exact Match
3	Two thirds to three quarters of people without [health] insurance in Rhode Island work.	True	Half-true	Close Match
4	The USA Freedom Act undercuts privacy because the phone records will be in the hands of the phone companies with hundreds of people available to look at the records, versus 20 or 30 people in the government.	Half-true	False	Wrong Match
5	The people who were running the budget, in the Corzine years, decided to steal from the unemployment trust fund. As did other Governors before. We simply dont do that, havent done it, and wont permit it.	Half-True	Barely-True	Close Match

The first step of our model includes a sequence encoding block. This block takes a sentence vector having dimension $[l, D_{input}]$ and performs the concerned operations on the data while giving as output $\overrightarrow{d}$ having dimension D_{output} in the process. The operations may vary depending on the nature and architecture of the model used in the encoding block. The sequence encoding block can be vanilla sequence models like LSTM or CNN. It might also involve the use of variants of these vanilla models like RCNN, Bi-LSTM, etc. We discuss the basic candidates for the sequence encoding block in section VII. It should be noted the the sequence encoding block can be replaced by any neural network architecture that satisfies the above conditions. The mathematical function of the block is given by equation 6

$$R(\overrightarrow{s_k}) = \overrightarrow{d_k} \tag{6}$$

where R denotes the operations carried out by the sequence encoding block, $\overrightarrow{s_k}$ is the sentence vector and $\overrightarrow{d_k}$ represents the encoded sentence vector.

In the second step, we refocus the sentence encoding (obtained as mentioned above) onto the speaker's credit history using a customized tensor product (CTP) as discussed in section V. The mathematical synopsis of this step can be defined as equation 7.

$$\overrightarrow{e_k} = CTP(\overrightarrow{d_k}, \overrightarrow{c_k}) \tag{7}$$

In the next step, a block containing dense layers is used to transform metadata one-hot encoded features (discussed in section VI-B) into a D dimension vector $\overrightarrow{y}_{sp_k}$ as given by equation 8

$$\overrightarrow{y}_{sp_k} = \text{activation}\left(\sum_{z=0}^{D-1}\sum_{a=0}^{m-1} sp_{k_a} * W_{az} + b_{az}\right) \tag{8}$$

where $\overrightarrow{sp}_k$ corresponds to profile vector of speaker obtained from metadata. sp_{k_a} is the a^{th} element of $\overrightarrow{sp}_k$. W_{az} and b_{az} is the weight and bias associated with input neuron a to output neuron z respectively.

For joining vectors there are two common approaches: dot product and concatenation. After suitable experimentation it was observed that simple concatenation was more effective. Thus, in the final step, we concatenate the transformed metadata features with credit features and refocused encoded features. The essence of the mathematical operation conducted in this part can be given by equation 9.

$$\overrightarrow{concat}_k = \overrightarrow{e}_k + \overrightarrow{c}_k + \overrightarrow{y}_{sp_k} \tag{9}$$

The $\overrightarrow{concat}_k$ thus obtained is passed through a number of dense layers and finally a softmax layer to get the output.

VIII. EXPERIMENTAL DETAILS

In this section, we discuss the various experimental settings we have used to evaluate our model. In the section VIII-A, we describe the various models used in sequence encoding block keeping all the other factors constant. In the section VIII-B, we vary the features used for making the classification of the news.

A. Sequence Encoding models

To measure the effectiveness of our curated features, a thorough experimentation has been conducted by varying the models used in sequence encoding block and the hyperparameters associated with it. The different sequence encoding models used are: LSTM, CNN, LSTM-CNN and RCNN.

1) LSTM Model: Long Short-Term Memory (LSTM) [23] is a refined recurrent neural network (RNN) based architecture having prowess to deal with long-term dependencies occurring in a time series data. For the LSTM model, the dimension of the input D_{input} is 768 and dimension of hidden layer is also 768. The number of time steps is equal to the length $l = 25$. The last hidden state is taken the final output having dimension $D_{output} = 768$.

2) CNN Model: Convolutional Neural Networks [16] are feed-forward spatial neural networks comprising of alternate layers of convolution and subsampling. In the CNN model, we take three 1D-convolution layers with size of input channels 768 (D_{input}) and the size of output channels 768. The size of three different kernels k_1, k_2 and k_3 used are 2, 5 and 8. 1-max pooling operation is, then, carried out on the outputs received from the kernels k_1, k_2, and k_3. The three output vectors are concatenated to give a final output vector Y having dimension $D_{output} = 2304$ (3×768).

3) LSTM-CNN Model: This method helps to combine LSTM and CNN by giving the output of the LSTM as input to our CNN model. The $\overrightarrow{s_k}$ of dimension 768 (D_{input}) is passed through the LSTM model with a hidden layer of 768 dimension and number of time steps $l = 25$. Thus, we obtain a vector of size (25, 768) denoted by $\overrightarrow{h} = [h_{w_1}, h_{w_2}, h_{w_3} \dots h_{w_25}]$. This $\overrightarrow{h}$ holds the essence of long term dependencies of the statement. This vector serves as input for CNN model (refer to section VIII-A2). It gives final output vector Y in the having dimension $D_{output} = 2304$ (3×768)

4) RCNN Model: In RCNN [18] model, first we apply a bi-directional recurrent model that imports significantly lesser noise as compared to conventional window based networks. It efficiently captures the contextual information without introducing significant amount of noise in data. Then, we use the max pooling to automatically determine the crucial features present in the data. Thus, RCNN uses the benefits of CNN and the RNN architecture model for better performance. For the RCNN model, $\overrightarrow{s_k}$ is the input vector of dimension $D_{input} = 768$ and number of time steps $l = 25$ is passed into a bi-directional LSTM model having hidden layer of size 768. The output from this model is concatenated with the initial input yielding a vector of dimension (25, 2304) $(2304 = 768 \times 2 + 768)$. This output is transformed to a vector of size (25, 768) by using a fully connected layer. The output of the fully connected layer is max-pooled to get the final output having dimension $D_{output} = 768$.

B. Features used for Classification

For the purpose of evaluating our features against the conventional features, we have varied the features used in the classification model. The variations include using only the word2vec [24] based features extracted from the statement of the speakers, simple BERT based features extracted from the statement of the speakers, speaker profile features combined with word2vec features, speaker profile features combined with BERT features and finally the BERT features combined with speaker profile features and speaker credibility features. For all these cases, LSTM-CNN model has been used due to its superior performance. The performances by these models can be seen in Table IV.

The learning rate for all the models is set to 0.00005 after appropriate hyperparameter tuning. A dropout of 0.5 has been used after every layer in each model to avoid overfitting. For other parameters, we have used default values of the PyTorch environment.[3] The batch size has also been varied to achieve better performances as can be seen in Table III.

[3]For training the model we utilise GeForce GTX 1080, having 12GB GPU RAM. Together with a 30 Core CPU having 3GB of System Memory per Core.

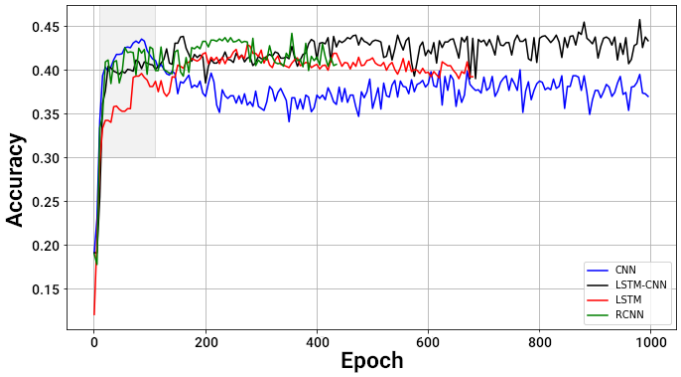

Figure 3. Accuracy Curves for Different Sequence Models

IX. Results and Discussion

The quality of features obtained by our technique can be observed in Table IV. The word2vec embeddings for statements yield 26.36% accuracy in the prediction of classes for these statements. We employ BERT embeddings instead of word2vec to achieve an improved performance accuracy of 27.18%. To further improve the performance, we incorporate metadata into the sentence embedding feature. A combination of metadata with BERT embedding again outperformed the combination of metadata with word2vec achieving an accuracy of 30.99% over 27.41%. In the final step, we further refined the features by appending the speaker's credibility to it by the means of credit vectors computed in section VI-C. The baseline model has been modified into a hybrid model as described in section VII for the exploitation of all three features. This leads to a significant improvement in performance to 46.80%.

This demonstrated that enhancing contextual features using BERT led to marginal improvement in performance. This performance was again marginally improved by using the speaker's profile in the form of metadata along with statement embeddings. However, the speaker's credibility played the most crucial role in predicting the legitimacy of the speaker's statement by boosting the accuracy of a benchmark result.

According to Table III, the highest validation accuracy of 46.80% was obtained from hybrid LSTM-CNN model with a batch size of 128. This model outperformed the baseline LSTM and CNN models by modest margins. LSTM gave a performance achieving validation accuracy of 42.78% owing to the temporal attributes of the model. Meanwhile, CNN achieved a performance of 44.28% on the validation set by exploiting spatial correlation present in the sentence. RCNN also managed to outperform the baseline models achieving an accuracy of 44.32. The validation curves for the sequence models are as shown in Fig. 3.

The examples numbered 1 and 2 in Table II are correctly identified as per the ruling of respective statements. However, the examples 3, 4 and 5 are closely matched or wrongly

Table III
ACCURACY COMPARISON AMONG DIFFERENT SEQUENCE ENCODING MODELS.

Model	Accuracy	Batch size
LSTM	0.427827	64
CNN	0.442835	32
LSTM-CNN	**0.468040**	**128**
RCNN	0.443182	128

Table IV
PERFORMANCE ANALYSIS OF DIFFERENT FEATURE MODELS

Features	Accuracy
Word2Vec	0.2636
BERT	0.2718
Word2Vec + MetaData	0.2741
BERT + MetaData	0.3099
BERT + MetaData + Credit	**0.4680**

matched against the ruling. Some of the observations on why this happens include the text normalization issues and excessive statement length apart from phrase and sentence level ambiguity related issues.

We have compared our results with the previous works on LIAR dataset as shown in Table V. Our model achieved better results than the previous techniques classifying the news into six classes of LIAR dataset. It is to be noted that our model performs six class classification instead of a binary classification segregating the news into fake and true news without determining its level of precision or falsification.

To further enhance the performance of the model, attention can be used to incorporate metadata into the model instead of using simple concatenation. The performance can be further boosted by improving the credit history vectors by using the information from multiple resources. Also, the credit history contains overall history of credibility of a speaker. Due to this reason, the model might get biased against a speaker who has poor overall credibility record but clean record in recent years. This limitation can be overcome by using weights varying with time to reduce the bias against any such speaker.

X. CONCLUSION

In this work, we propose a predictive model that can efficiently segregate the fake news from the real ones. Since, most fake news are tweaked version of some actual news story, detecting it can be a challenging task. We use a hybrid sequence encoding model coupled with some specific mathematical operations to reinforce additional inputs to supply more information to contextual data. This additional information can be in two forms:

1) Metadata encapsulating information about the speaker: the job of the speaker, political party affiliation of the speaker, state of residency of the speaker, topic on which statement is made and the location of the speech
2) Credit vector portraying the frequency at which the speaker is known to tell a lie.

Table V
PERFORMANCE OF PREVIOUS WORKS ON THE LIAR DATASET

Model	Accuracy
Hybrid-CNN [8]	0.274
MMFD [25]	0.348
LSTM-Attention [10]	0.415
Bi-LSTM [26]	0.415
CreditLSTM [12]	0.457
BERT + MetaData + Credit	**0.4680**

Experimentation conducted on the LIAR dataset shows that the inclusion of the additional data can assist significantly in improving the performance of the predictive model. It can be interpreted as the speaker's tendency to speak the truth or lie remarkably depends on the profile of the speaker and his/her affinity to speak the truth or lies in general (credit history). For example, a political speaker might try to mold the truth in a political demonstration for his/her political motivation. The results obtained by our model resonates with this hypothesis by achieving improved performances over previous state-of-the-art approaches for the LIAR dataset.

REFERENCES

[1] V. L. Rubin, N. Conroy, Y. Chen, and S. Cornwell, "Fake news or truth? using satirical cues to detect potentially misleading news," in *Proceedings of the second workshop on computational approaches to deception detection*, 2016, pp. 7–17.

[2] H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," *Journal of economic perspectives*, vol. 31, no. 2, pp. 211–36, 2017.

[3] X. Zhou and R. Zafarani, "A survey of fake news: Fundamental theories, detection methods, and opportunities," *ACM Computing Surveys (CSUR)*, 2020.

[4] S. Cohen, J. T. Hamilton, and F. Turner, "Computational journalism," *Communications of the ACM*, vol. 54, no. 10, pp. 66–71, 2011.

[5] B. M. DePaulo, J. J. Lindsay, B. E. Malone, L. Muhlenbruck, K. Charlton, and H. Cooper, "Cues to deception." *Psychological bulletin*, vol. 129, no. 1, p. 74, 2003.

[6] Y. Wang, F. Ma, Z. Jin, Y. Yuan, G. Xun, K. Jha, L. Su, and J. Gao, "Eann: Event adversarial neural networks for multi-modal fake news detection," in *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining*, 2018, pp. 849–857.

[7] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-F. Wong, and M. Cha, "Detecting rumors from microblogs with recurrent neural networks," 2016.

[8] W. Y. Wang, ""liar, liar pants on fire": A new benchmark dataset for fake news detection," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2017, pp. 422–426.

[9] S. Gottipati, M. Qiu, L. Yang, F. Zhu, and J. Jiang, "Predicting user's political party using ideological stances," in *International Conference on Social Informatics*. Springer, 2013, pp. 177–191.

[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)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, Nov. 2017, pp. 252–256. [Online]. Available: https://www.aclweb.org/anthology/I17-2043

[11] M.-A. Abbasi and H. Liu, "Measuring user credibility in social media," in *International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*. Springer, 2013, pp. 441–448.

[12] A. Kirilin and M. Strube, "Exploiting a speaker's credibility to detect fake news," in *Proceedings of Data Science, Journalism & Media workshop at KDD (DSJM'18)*, 2018.

[13] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[15] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," 1999.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[17] S. Song, H. Huang, and T. Ruan, "Abstractive text summarization using lstm-cnn based deep learning," *Multimedia Tools and Applications*, vol. 78, no. 1, pp. 857–875, 2019.

[18] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[19] M. Marcus, B. Moyls *et al.*, "Transformations on tensor product spaces," *Pacific J. Math*, vol. 9, no. 4, pp. 1215–1221, 1959.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186.

[21] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–394, 1999.

[22] S. L. Zabell, "The rule of succession," *Erkenntnis*, vol. 31, no. 2-3, pp. 283–321, 1989.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] 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*, 2013, pp. 3111–3119.

[25] H. Karimi, P. Roy, S. Saba-Sadiya, and J. Tang, "Multi-source multi-class fake news detection," in *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 1546–1557. [Online]. Available: https://www.aclweb.org/anthology/C18-1131

[26] M. K. Balwant, "Bidirectional lstm based on pos tags and cnn architecture for fake news detection," in *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 2019, pp. 1–6.