A NOVEL APPROACH FOR DETECTING FAKE NEWS THROUGH MULTI LAYER PERCEPTRON INCLUDING SENTIMENTS

[1]Devireddy Pravallika, [2]S. Kanthi Kiran

[1]PG Student of Department of IT&CA, AU College of Engineering(A), Visakhapatnam, AP, India,

[2]Associate Professor, Department of IT, GVP College of Engineering(A), Visakhapatnam, AP, India.

Abstract- Fake news is a kind of false information to intentionally mislead or manipulate public opinion, through conventional mass media and ongoing social media. In recent years, due to explosive growth of online social media, fake news for different political agenda and commercial gains has been coming out in wide spread online. The main objective of the proposed method is to effectively detect fake news and prevent its diffusion on social media which has gained much attention in recent years. Here we also include latent sentiments hidden in user comments which can help to distinguish fake news from reliable content. We incorporate these latent sentiments into deep embedding framework (Multi-Layer Perceptron) for detecting fake news. First, we use multi modal networks to deal with heterogeneous data modalities. Second, to learn semantically meaningful spaces per data source, an adversarial mechanism is adopted. Third, we characterize a novel regularization loss to bring embeddings of relevant pairs closer. The real-world dataset used is PolitiFact, demonstrates the effectiveness in detecting fake news.

Keywords: Adversarial Mechanism, Fake news, Latent Semantics, Multi-modal networks, Social Media.

I. INTRODUCTION

The main objective of this paper is to effectively detect fake news and prevent its diffusion on social media. Existing methods for detecting fake news can be generally categorized into two categories based on the heterogeneity of the data i.e., single modal-based methods and multi modal based methods. In Single Modal Based methods we will consider often textual information which includes contents, profiles, description. Fake news often comes with multi-modality data including manipulated images, fake videos or user comments.

In addition to the issue of modality, another important idea is to exploit the latent sentiment in user comments. Although user's viewpoint has been proved to be useful in fake news detection [2], there are few studies on the impact of user's sentiment. User's comments such as "I agree. she is a rock star" or "No, it's a fake news story specifically targeting 'conservative readers" may potentially add/remove different degrees of credibility to the news in question. Hence, towards the detection of fake news, we propose to apply the sentiment analysis for both user comments as well as multi-modal fake news data.

Users opinions or sentiments towards posts or products in social media have been demonstrated to be very effective for many social media mining tasks. Detecting user sentiments or stances has become a popular task. In literature [4] authors conduct user's belief classification and in literature [5] authors conduct stance detection. Zhang et al. [3] focus on the news stance detection. The proposed model takes the headline and body of an article, and generates the probabilities of four news stances including "agree", "disagree", "discuss" and "unrelated". The authors use ranking-based to address the problem brought by classification-based algorithms that a clear distinction exists between any two stances. In literature [6], the authors predict the stance of a set of texts representing facts with respect to a given claim by using end-to-end memory networks. As sentiment features have shown promising results in improving the performance of news stance detection, we introduce sentiment features into the fake news detection task.

For incorporating user's sentiment into a detection procedure with multi-modal data, the first step is to represent a user. Each user may comment or "like" a particular type of news. Such a representation can be measured by the correspondence between user's historical interest and type of news. However, this problem is technically difficult for two reasons. On one hand, the learned features of user's representation are usually high dimensional and sparse, which cannot be set up by conventional methods. On the other hand, as each modality has an intrinsically different distribution, it is challenging to fuse user's representation with others. For example, a user's sentiment is represented as sparse while the image feature is naturally dense, causing a mismatch.

Overcoming these challenges, in this paper, we present a novel method, named as Sentiment-Aware Multi-modal Embedding, with the emphasis on both sentiment and multi-modality. We propose to use an end-to-end deep architecture to reduce the heterogeneity introduced by multimodal data and capture the representation of user's sentiment better. To be specific, first, we use different networks to deal with the triplet relationship among news publishers, users, and news. Second, an adversarial mechanism is introduced to preserve the semantic similarity and enforces the representation consistency between different modalities. Third, we model user's sentiment and incorporate it into the proposed framework.

Table 1: Sentiment Polarity distribution under news

	Negative	Neutral	Positive
Politifact Fake News	12.6	45.3	14.6
Real News	8.4	75.9	13.2
Gossipcop Fake News	9.8	69.2	21
Real News	8	74	16.7

Our main contributions are as follows:

- We propose an end-to-end deep framework to integrate different features of news contents for fake news detection. An adversarial mechanism is added to preserve semantic relevance and the representation consistency across different modalities.
- We validate the effectiveness of user sentiment through statistical analysis and use users' sentimental polarities to facilitate fake news detection.
- We empirically demonstrate that our proposed method, significantly outperforms five state-of-the-art baselines in detecting fake news on social media using two real-world benchmark datasets.

II. DATASET

Here we considered two widely used multi-modal fake news detection datasets, i.e., PolitiFact and GossipCop, which are publicly available from a fake news dataset repository FakeNewsNet[7]. For both datasets, each news entity contains news content, corresponding images, users' retweets/replies and news profile (source, publisher and keywords). Each news has 0 to 1,000 user comments. Some users didn't leave a comment when they retweet, so we excluded such kind of user data.

- PolitiFact is a fact-checking website that targets on political news. It rates the authenticity of claims by elected officials and others. The two datasets are crawled from Twitter in order to get users' comments.
- GossipCop is a fact-checking website for celebrity reporting. It investigates the credibility of entertainment stories on magazines, newspapers and social media, to ascertain whether they are real or not.

III. SENTIMENT ANALYSIS TOWARDS FAKE AND REAL NEWS

Intuitively, the comments under fake news can be roughly divided into three classes:

1) Agree (Users who believe in news)
2) Discuss (Users who doubt the authenticity of the news)
3) Disagree (Users who do not believe in news)

Usually, the first and third types of comments contain polarized emotions ("Negative" and "Positive"), which can be seen from User1 and User3 in Figure 1. The second type of comments will not contain such strong emotions. The sentiment is more neutral in skeptical comments or discussions. Here we perform the sentiment analysis on the user's comments with VADER2 [8], which is a lexicon and rule-based tool to predict the sentiment expressed on social media. For each news piece, we obtain all the replies for this news and apply VADER to predict the sentiment as negative, neutral or positive. As can be seen from Table I, users' comments under fake news often contains more sentiment polarities and are less neutral.

IV. PROPOSED METHOD

In multi-modal embedding, we will have four objects to consider such as image of the news, content of the news, profile (Publisher) and comments of the user. Therefore, this multimodal data consists of three modalities. Let us assume that we have N training pairs $D = \{T_i; r_i\}_{i=1}^{N}$ in which

- T_i denotes news i and
- r_i belongs to {0; 1} denotes its ground truth label.

Let $T_i = (x_i , y_i , z_i)$

- X_i Denotes the feature vectors of image modality
- Y_i Denotes the feature vectors of text modality
- Z_i One hot code of news profile related to news

In addition, we are also given a similarity matrix S, where S_{ij} evaluates the similarity of news i and news j. The similarity is defined by the shared user's sentiment. For example, we can say that news i and news j are similar if they share multiple sentiment words. We first introduce how to learn the latent news presentation from the multi-modal news data by learning a joint embedding function $f(T_i)$ map the news to space RM, where different modalities are distributed consistently. To preserve the similarity matrix S, the distance between embedding vectors $h_i = f(T_i)$ and $h_j = f(T_j)$ should be small if S_{ij} is relatively large. Thus, a hybrid similarity loss is proposed to embed the user's sentiment into the model. The objective is to maximize the similarity between similar news triplets and minimize it between all dissimilar news triplets. Finally, once each data source is mapped to RM, we use the embedding vector hi to predict the news label r_i.

3.1. Multi-Modal Feature Extractor

We built the image network based on VGGNet [9], which is pre-trained on the ImageNet database [10]. To fit CNN into our model, we reserve the first seven layers and replace the eighth layer by a layer with R nodes, fch_i. As for the text network, we use GloVe [11] to process text y, in order to capture the complex characteristics of word use (e.g., syntax and semantics). Acquired text

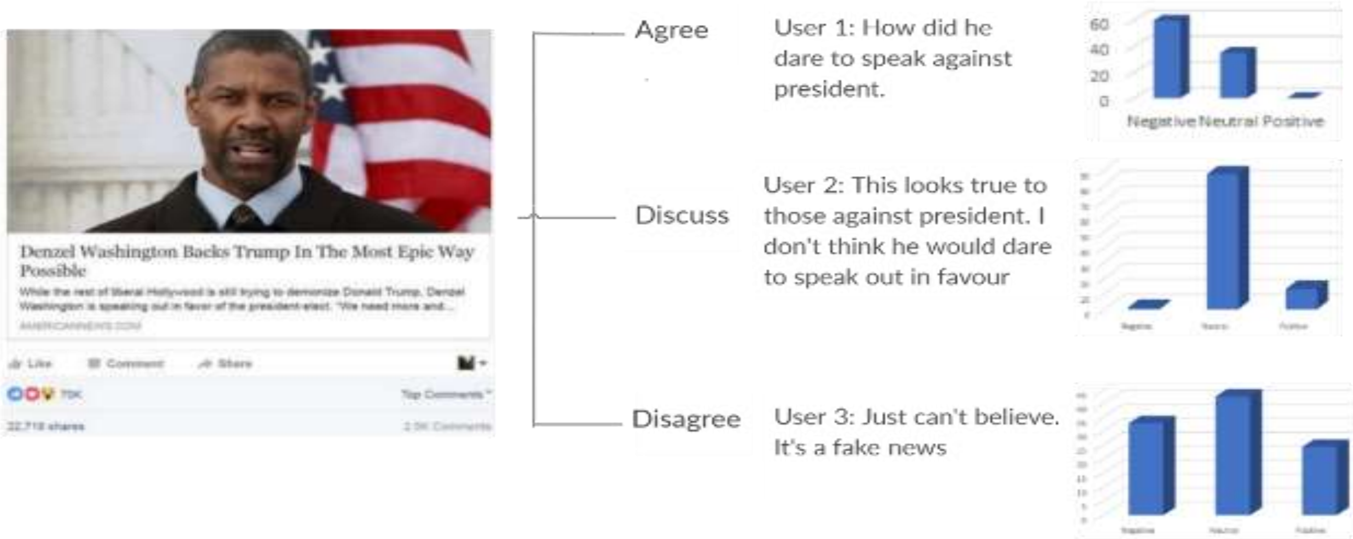

Fig 1: Sentiment polarity distribution of different stances in users' comments

representation is used as the input of the text network. We then adopt the Multi-Layer Perceptron (MLP) comprising two fully connected layers. The second layer fch_t has R hidden units, which can transform the network activation to R-dimensional representation. As for profile information, the features are discrete values such as topic. Therefore, we use the one-hot encoding to represent the profile z, and feed it to a two-layer fully-connected MLP, and get the representation fch_p. As for the adversarial networks, we built the discriminator networks by using a three-layer feed-forward neural network.

To integrate the three networks mentioned above, a fully connected layer with M hidden units, which takes the representations of three networks as input, is added on top of the architecture. We denote the multi-modal feature extractor for news i as $f^{(v)}(T^{(v)}; \theta_a) \in R^M$, which corresponds to the output of the hybrid deep network for multi-modal correlation embedding. Here, θ_a is the network parameter to be learned.

3.2. Adversarial Learning

With the above network, however, different modalities are usually distributed inconsistently, which is not beneficial if we use the concatenation for fake news detection. In order to bridge this modality gap, we introduce an adversarial learning mechanism. We use two discriminators for image and profile modalities to investigate their distributions. For the image (profile) discriminator, the inputs are image (profile) features and text features obtained from the feature extractor, and the output is a binary label, either "0" or "1".

Specifically, we denote the modality label for the textual feature that has been generated from text network as "1" and define the modality label for image (profile) semantic modality features generated from image network (profile network) as "0".

We feed the outputs of image and text network into one discriminator and feed the outputs of profile and text networks into the other discriminator. The loss functions of the two discriminators can be defined as L^i_a and L^p_a. The two discriminators act as the two adversaries while we are training our model. The loss function L^i_a can be written as:

$$\min_{\theta c} L^i_a = \sum_{j=1}^{2N} D^{i,t}(fch_j^* - d_j^*) \qquad (1)$$

where fch_j^* is semantic features obtained from image network or text network, the modality label is d_j.

Specifically, we have $d_i^j = 0$ denoting the modality label for image and $d^t_j = 1$ denoting the label for text. The result of Eqn. (1) is that the discriminator acts as a binary classifier $D^{i;t}(fch_j^*, \theta_c)$, classifying the input features into class "1" and class "0". Similarly, we have L^p_a.

3.3. Modeling Sentiment Correlation

To preserve the similarity information, we will make the learned joint embeddings maximally preserve the similarity information, we propose a novel hybrid similarity loss by considering such two issues:
(1) entity triplets with lower similarity should be separated and have discriminative embeddings.
(2) entity triplets should have similar embeddings only if they are similar in the original feature spaces.

To address the first issue, we propose the Graph Affinity Metric between news i and news j. The Graph Affinity Metric is defined as follows:

Definition 1: Let G_{ij} denotes the similarity of sentiment polarity distribution between the comments of news i and j. We can define the Graph Affinity Metric between two news as G_{ij}. Then, we define the Local Similarity Metric to ensure the local similarity in each news to ensure the second issue above.

Definition 2: The Local Similarity Metric $L_{i,j;m}$(m = 1; 2; 3) of each modality involves the local similarity information.

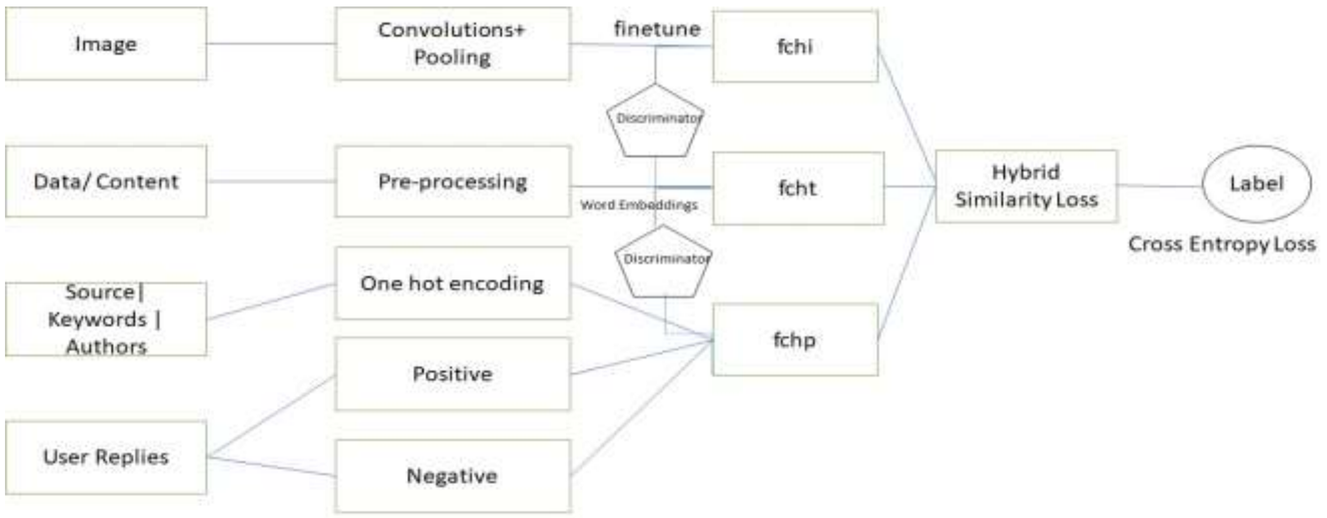

Fig 2: Multi-modal Embedding accepts input in a triplet of news publisher, user and news, and then processes them through deep network.

On modality x, we have

$$L_{ij,1}^{(v)} = \begin{cases} 1, & \text{if } \mathbf{x}_i \in \mathbf{N}_k(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in \mathbf{N}_k(\mathbf{x}_i) \\ 0. & \text{otherwise} \end{cases}$$

where $N_k(.)$ denotes the set of k-nearest neighbors. Similarly, we have $L_{ij;2}$ and $L_{ij;3}$ defined on modalities y and z respectively. To maintain the similarity between entities and preserve the local structural information in the common embedding space, we propose a hybrid similarity loss which ensures the learned embedding space meaningful:

$$\min_{\theta a} L_c = ½ \sum_{i,j=1}^{N} S_{i,j}||h_i - h_j||^2_2 \qquad (2)$$

where $S_{ij} = G_{ij} + L_{ij;1} + L_{ij;2} + L_{ij;3}$.

3.4. Fake News Detector

Here we introduce how to detect fake news by using the M-dimensional embedding. We use a fully connected layer with softmax, which is shown in Figure 2. Each network takes embedding vectors h_i of news i as input. We have a training set $\{r_i\}^N_{i=1}$, where r_i € {0; 1} denotes the ground truth label of news i. The goal is to find a set of prediction function g, such that the label for any news i can be predicted. We denote the fake news detector as $g^{(v)}(f^{(v)}(T^{(v)}{}_i; \theta_a); \theta_b)$ € R, where θ_b is the network parameter of the network for fake news detector. Assume the ranking score is modeled as $\hat{}r_i = [\hat{}r_{i;0}; \hat{}r_{i;1}]$, with $\hat{}r_{i;0}$ and $\hat{}r_{i;1}$ indicate the predicted probability of label being 0 (real news) and 1 (fake news) respectively. r_i denotes the ground truth label of news. Thus, for each news, the goal is to minimize the cross-entropy loss function as follows:

$$\min_{\theta a, \theta b} L_q = -r_i \log(\hat{}r_{i;1}) - (1 - r_i) \log(1 - \hat{}r_{i;0}) \qquad (3)$$

3.5. The Proposed Method

During the training, the feature extractor and the fake news detector work together to minimize the detection loss Lq. Simultaneously, the feature extractor tries to fool the discriminator to get distribution agreement for different modalities by maximizing the adversarial loss $L^i_{a\,i}$ and L^p_a. The final objective function of the proposed method is:

$$Jg = Lc + \beta Lq$$
$$Ja = L^i_a + L^p_a \qquad (4)$$

where β is a penalty parameter for trading off the relative importance of multi-modal correlation and news label. We set β = 1 based on empirical study. If we put them together, we can obtain:

$$(\theta_a, \theta_b) = \arg \min_{\theta a, \theta b} J_g(\theta_a, \theta_b) - J_a(\hat{}\theta_c)$$
$$\theta_c = \arg \max_{\theta c} J_g(\hat{}\theta_a, \hat{}\theta_b) - J_a(\hat{}\theta_c) \qquad (5)$$

All the parameters in the network are learned through RMSprop, which has widely used among existing methods. It is an adaptive learning rate method which divides the learning rate by an exponentially decaying average of squared gradients.

V.RESULTS

The below is the screenshot for the number of unique tokens determined from text and metadata of our dataset. We require these unique tokens to evaluate whether the title and the metadata are related to each other and we compare these word tokens with the images to check for validity and pass them through adversarial mechanism. These word tokens are passed through glove network.

```
[ ] word_index = tokenizer.word_index
    print('Total %s unique tokens.' % len(word_index))

    Total 9174 unique tokens.
```

Fig 3: Number of Unique Tokens

Below is the screenshot of "Glove" with number of unique tokens which are compared with the unique tokens shown in Figure 3.

```
[22] GLOVE_DIR = "."
     embeddings_index = {}
     f = open(os.path.join(GLOVE_DIR, 'glove.6B.100d.txt'))
     for line in f:
         values = line.split()
         word = values[0]
         coefs = np.asarray(values[1:], dtype='float32')
         embeddings_index[word] = coefs
     f.close()

     print('Total %s word vectors.' % len(embeddings_index))

     Total 400000 word vectors.
```

Fig 4: Word vectors of Glove

Below is the screenshot of data processed after Multi Modal Feature Extractor which takes input in a triplet of news image, content and profile. Here we built Image network based on VGGNet, which is trained on ImageNet database.

```
base_model = VGG16(include_top=True, weights='imagenet', input_tensor=input_tensor,
                   input_shape=None, pooling=None, classes=1000)
im = base_model.output
```

Fig 5: Using VGGNet trained on ImageNet database

```
Model: "generator"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_4 (InputLayer)            (None, 224, 224, 3)  0
__________________________________________________________________________________________________
block1_conv1 (Conv2D)           (None, 224, 224, 64) 1792        input_4[0][0]
__________________________________________________________________________________________________
block1_conv2 (Conv2D)           (None, 224, 224, 64) 36928       block1_conv1[0][0]
__________________________________________________________________________________________________
block1_pool (MaxPooling2D)      (None, 112, 112, 64) 0           block1_conv2[0][0]
__________________________________________________________________________________________________
block2_conv1 (Conv2D)           (None, 112, 112, 128 73856       block1_pool[0][0]
__________________________________________________________________________________________________
block2_conv2 (Conv2D)           (None, 112, 112, 128 147584      block2_conv1[0][0]
__________________________________________________________________________________________________
block2_pool (MaxPooling2D)      (None, 56, 56, 128)  0           block2_conv2[0][0]
__________________________________________________________________________________________________
block3_conv1 (Conv2D)           (None, 56, 56, 256)  295168      block2_pool[0][0]
__________________________________________________________________________________________________
block3_conv2 (Conv2D)           (None, 56, 56, 256)  590080      block3_conv1[0][0]
__________________________________________________________________________________________________
block3_conv3 (Conv2D)           (None, 56, 56, 256)  590080      block3_conv2[0][0]
__________________________________________________________________________________________________
block3_pool (MaxPooling2D)      (None, 28, 28, 256)  0           block3_conv3[0][0]
__________________________________________________________________________________________________
block4_conv1 (Conv2D)           (None, 28, 28, 512)  1180160     block3_pool[0][0]
__________________________________________________________________________________________________
block4_conv2 (Conv2D)           (None, 28, 28, 512)  2359808     block4_conv1[0][0]
__________________________________________________________________________________________________
block4_conv3 (Conv2D)           (None, 28, 28, 512)  2359808     block4_conv2[0][0]
__________________________________________________________________________________________________
block4_pool (MaxPooling2D)      (None, 14, 14, 512)  0           block4_conv3[0][0]
```

Fig 6: Output of Multi Modal Feature Extractor

```
D_in = Input(shape=(EMBEDDING_DIM,))
D, D_out = get_discriminative(D_in)
D.summary()
```

```
Model: "discriminator"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_7 (InputLayer)         (None, 100)               0
_________________________________________________________________
dense_5 (Dense)              (None, 100)               10100
_________________________________________________________________
leaky_re_lu_1 (LeakyReLU)    (None, 100)               0
_________________________________________________________________
dropout_1 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_6 (Dense)              (None, 100)               10100
_________________________________________________________________
leaky_re_lu_2 (LeakyReLU)    (None, 100)               0
_________________________________________________________________
discriminator_output (Dense) (None, 3)                 303
=================================================================
Total params: 20,503
Trainable params: 20,503
Non-trainable params: 0
_________________________________________________________________
```

Fig 7: Output of Discriminator

The below is the screenshot for the output of adversarial learning which will take Figure 6 and Figure 7 outputs as their inputs. Figure 8 is the output of adversarial learning which takes the input from generator and discriminator.

```
Model: "model_1"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_8 (InputLayer)            (None, 224, 224, 3)  0
__________________________________________________________________________________________________
input_9 (InputLayer)            (None, 500)          0
__________________________________________________________________________________________________
input_10 (InputLayer)           (None, 164)          0
__________________________________________________________________________________________________
generator (Model)               [(None, 2), (None, 1 138785446   input_8[0][0]
                                                                 input_9[0][0]
                                                                 input_10[0][0]
__________________________________________________________________________________________________
discriminator (Model)           (None, 3)            20503       generator[1][1]
==================================================================================================
Total params: 138,805,949
Trainable params: 138,785,446
Non-trainable params: 20,503
__________________________________________________________________________________________________
```

Fig 8: The Output of Adversarial Learning

```
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
Epoch #1: Generative Loss: [0.7087215, 0.83736205, 0.62498534], Discriminative Loss: [0.084512345, 0.8282548]
Training D...
Training GAN...
Epoch #2: Generative Loss: [0.61817753, 0.48290402, 0.56988716], Discriminative Loss: [0.10498618, 0.9430894]
Training D...
Training GAN...
Epoch #3: Generative Loss: [0.5125153, 0.35611898, 0.47690344], Discriminative Loss: [0.17520723, 0.699187]
Training D...
Training GAN...
Epoch #4: Generative Loss: [0.41723853, 0.3218471, 0.3850538], Discriminative Loss: [0.20064034, 0.6585366]
Training D...
Training GAN...
Epoch #5: Generative Loss: [0.2850623, 0.31134325, 0.25392798], Discriminative Loss: [0.19931139, 0.6666667]
Training D...
Training GAN...
Epoch #6: Generative Loss: [0.18931243, 0.31704718, 0.1576077], Discriminative Loss: [0.1960721, 0.6666667]
Training D...
Training GAN...
Epoch #7: Generative Loss: [0.11336984, 0.33167323, 0.08020251], Discriminative Loss: [0.19149835, 0.6666667]
Training D...
Training GAN...
Epoch #8: Generative Loss: [0.083306685, 0.33654258, 0.049652427], Discriminative Loss: [0.18218048, 0.6666667]
```

Fig 9: Generative and discriminative Loss after Modelling Sentiment Correlation

```
y = np.zeros((L, 3))
y[:, 0] = 1 #fool D that all texts are images
g_loss.append(GAN.train_on_batch([x_img_train, x_train, x_meta_train], [y, y_train]))

y[:, 2] = 1 #fool D that all texts data are meta data
g_loss.append(GAN.train_on_batch([x_img_train, x_train, x_meta_train], [y, y_train]))

if verbose and (epoch + 1) % v_freq == 0:
  print("Epoch #{}: Generative Loss: {}, Discriminative Loss: {}".format(epoch + 1, g_loss[-1], d_loss[-1]))
```

Fig 10: Application of proposed Method

The output from the figure 8 will be sent as input to the Hybrid similarity loss and then for Modelling sentiment correlation. From figure 9, i.e., Here we maintain the similarity between entities and preserve the local structural information in the common embedding space.

To get the actual output i.e., Figure 9 during training the feature extractor and the fake news detector work together to determine to minimize the detection loss. Simultaneously, the feature extractor tries to fool the discriminator to get distribution agreement for different modalities by maximizing the adversarial loss.

Here we used Generative adversarial Learning (GAN) which has a generator model and discriminator model. GAN achieves the realism by pairing a generator and the discriminator. Generator tries to fool the discriminator and the discriminator tries to keep from being fooled.

VI. CONCLUSION AND FUTURE WORK

We investigate a novel problem of exploring sentiments for fake news detection with multi-modal data. We first use statistical analysis to test the hypothesis in order to validate the effectiveness of user's sentiment. Then, we propose a new deep multi-modal embedding architecture for fake news detection, which unifies multi-modal data with adversarial learning and incorporates user's sentiment. The experimental results demonstrate the effectiveness of our method as well as the roles of user's sentiment in fake news detection. In addition, we also examine the necessity of each module in the proposed method and thus test the fusion network proposed. To improve the quality of information, there are several interesting directions that need further investigation. First, to mitigate the problem of fake news a better, extending our method to be able to do the early detection is important yet challenging. Second, most of current fake news detection methods uniquely focus on the detection. However, in addition to the detecting fake news, being able to "explain" why one is fake news is equally important.

REFERENCES

[1] SAME: Sentiment-Aware Multi-Modal Embedding for Detecting Fake News by Limeng Cui, Suhang Wang, Dongwon Lee, The Pennsylvania State University, PA, USA.
[2] Z. Jin, J. Cao, Y. Zhang, and J. Luo, "News verification by exploiting conflicting social viewpoints in microblogs." in AAAI, 2016, pp. 2972–2978.
[3] Q. Zhang, E. Yilmaz, and S. Liang, "Ranking-based method for news stance detection," in Companion of the The Web Conference 2018 on The Web Conference 2018. International World Wide Web Conferences Steering Committee, 2018, pp. 41–42.
[4] V. Qazvinian, E. Rosengren, D. R. Radev, and Q. Mei, "Rumor has it:Identifying misinformation in microblogs," in EMNLP. Association for Computational Linguistics, 2011, pp. 1589–1599.
[5] P. Sobhani, S. Mohammad, and S. Kiritchenko, "Detecting stance in tweets and analyzing its interaction with sentiment," in Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics, 2016, pp. 159–169.
[6] M. Mohtarami, R. Baly, J. Glass, P. Nakov, L. M`arquez, and A. Moschitti, "Automatic stance detection using end-to-end memory networks," in NAACL-HLT, vol. 1, 2018, pp. 767–776.
[7] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, and H. Liu, "Fakenewsnet: A data repository with news content, social context and dynamic information for studying fake news on social media," arXiv preprint arXiv:1809.01286, 2018.
[8] C. H. E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," in ICWSM, 2014.
[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.
[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein et al., "Imagenet large scale visual recognition challenge," International Journal of Computer Vision, vol. 115, no. 3, pp. 211–252, 2015.
[11] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in EMNLP, 2014, pp. 1532–1543.
[12] N. J. Conroy, V. L. Rubin, and Y. Chen, "Automatic deception detection: Methods for finding fake news," in Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community. American Society for Information Science, 2015, p. 82.
[13] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," ACM SIGKDD Explorations Newsletter, vol. 19, no. 1, pp. 22–36, 2017.
[14] Z. Jin, J. Cao, Y. Zhang, J. Zhou, and Q. Tian, "Novel visual and statistical image features for microblogs news verification," IEEE transactions on multimedia, vol. 19, no. 3, pp. 598–608, 2017.