MULTIMODAL FOR MISINFORMATION DETECTION

Pinyu Chen
Client: Dr. Zois Boukouvalas
American University

ABSTRACT

With the improvement of society, more and more methods are developed to help people extract daily information, especially social media. For instance, 6,000 tweets are posted on Twitter every second as of 2022. If we do the math, it is equivalent to 350,000 tweets per minute, 500 million tweets daily, and 200 billion tweets yearly (Ruby, 2022). With this among of information that produces every day, detecting misinformation becomes more and more significant. However, due to the great number of information released every day, distinguishing the “real” and “fake” information becomes a challenging task. Using deep learning is a way to detect misinformation in social media, however, the format of the information is not only text or comments but also images. This makes detection become even more difficult. Yet, to detect the text and image simultaneously, we can create a fusion model to make the detection more effective.

Keywords: Misinformation Detection, Data Fusion, Independent Vector Analysis, Multi-modal Learning, Deep Learning

1. INTRODUCTION

With the advancement of social media innovations, there has been an essential alteration in how data is transmitted, shared, and engendered. The proliferation of data, basically deception, gets particularly critical amid high-impact occasions such as pandemics, characteristic catastrophes, fear-monger assaults, periods of political move or turmoil, and money-related flimsiness.

Model fusion provides a superb solution to this complicated task, especially by using early fusion. Early fusion strategies give compelling arrangements for multi-modal learning since this method extracts the most represented features from the different modalities before attempting to classify them.

In addition, late fusion strategies also provide a confident solution for multi-modal learning since this method extracts the possibility from the different modalities after attempting to conduct the classification. To sum up, fusion modality has already been proven that can make the detection of misinformation become more efficient.

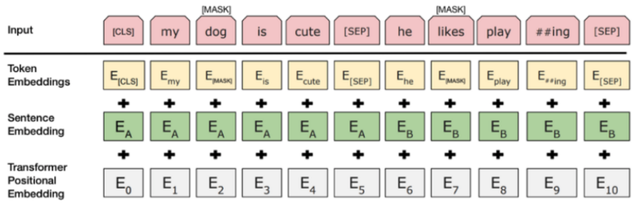
2. MATERIALS AND METHODS

2.1 DATASET

For this research, we used the data from tweet 2016, which included 9142 observations in training datasets and 796 observations in testing datasets, each dataset contains text, image, and label. The original dataset is labeled as “real” or ”fake” for each observation. Tweets labeled as ”fake” incorporate any post that offers mixed media substance that does not reliably speak to the occasion it alludes to. This incorporates substance from a past occasion reposted as being captured for a right now unfurling comparable event, a setting that has been deliberately controlled, or mixed media substance distributed with a wrong claim almost the portrayed occasion. For the image information, we pre-process each picture by resizing the pictures to 224 x 224 pixels and normalizing them. To get ready the information for utilization in include extraction, we clean the content information by expelling emoji characters, halt words, URLs, time stamps, and Twitter handles, selecting accentuation, and normalizing the content information by using lowercase content. In addition, minimize multi-spaces to one space, lemmatize the content, and keep as it were words that are more noteworthy than two characters long. Moreover, we removed observations with more than 512 tokens long after cleaning to plan for future forms. Of the training data tweets, 5,129 and 4,013 are considered fake and real. Of the test data tweets, 467 and 329 are considered fake and real.

3. FEATURE EXTRACTION BY BERT AND VGG-16

From the final, original training and test text and image datasets, we extract the features of text and image in order to classify tweet records as real or fake. We utilize various methods to create the features and evaluate each feature’s impact on classification accuracy. For the text features, we utilize pre-trained BERT (Bidirectional Encoder Representations from Transformers), which is a recent paper published by researchers at Google AI Language.



Source: BERT [Devlin et al., 2018], with modifications

Fig. 1: Illustration of BERT

A [CLS] token is inserted at the beginning of the first sentence and a [SEP] token is inserted at the end of each sentence. A sentence embedding indicating Sentence A or Sentence B is added to each token. Sentence embeddings are similar in concept to token embeddings with a vocabulary of 2. A positional embedding is added to each token to indicate its position in the sequence. The concept and implementation of positional embedding are presented in the Transformer paper.

And the pre-processed image tensors through the pre-trained VGG-16 model extract the first fully connected layer with 4,096 (224 * 224) hidden units produced by the VGG-16 model to utilize as the image features. VGG-16 is the abbreviation of the Visual Geometry Group of Oxford University, UK. The main contribution is to use more hidden layers, a large number of image training, and improve the accuracy to 90%.

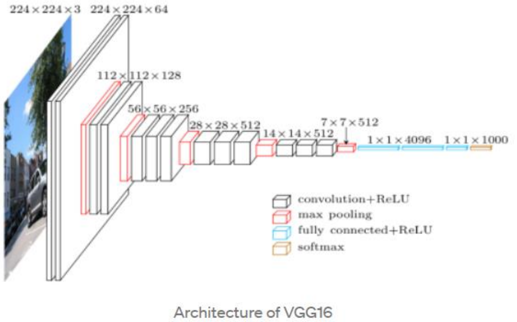
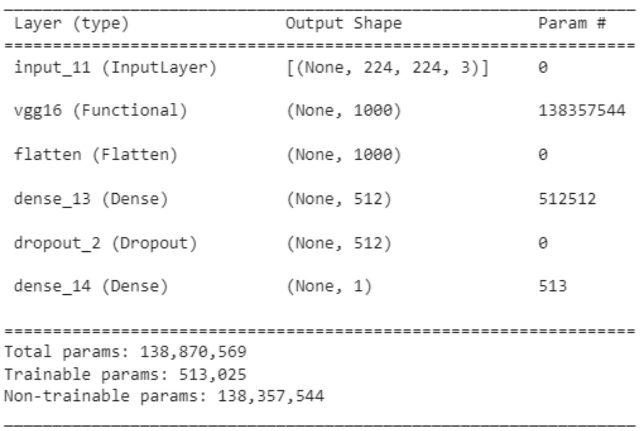


Fig. 2: Structure of VGG-16 (13 convolutional layers and 3 fully connected layers).

4. IMAGE MODEL

To achieve the best value of accuracy in the image model, we compared 4 different image models with sigmoid activation in the last layer for binary classification, Model 1 is a base pre-trained Model VGG-16, Model 2 is a base pre-trained model VGG-16 combined with one layer of relu activation, and one dropout layer, Model 3 is a base pre-trained Model VGG-16 with three layers, which are one for relu activation, one for Batch Normalization, and one for dropout and Model 4 is a base pre-trained Model VGG-16 with SELU activation (lecun_normal) and alpha dropout. The parameter of neural is 512, the early stopping monitor on validation loss with patience equal to 2, and for the learning strategy, we used exponential scheduling to make sure the learning rate will decrease as the loss function converges. For the model compilation, we selected binary cross entropy to loss, optimizer as Adam for the first 3 models Nadam for model 4 and metrics with accuracy.

The evaluation accuracy of each different model on testing sets is 0.45, 0.72, 0457, and 0.634. The model with base pre-trained model VGG-16 combined with one layer of relu activation, and one dropout layer obtained the highest accuracy. The plot below is the structure of Model 2.



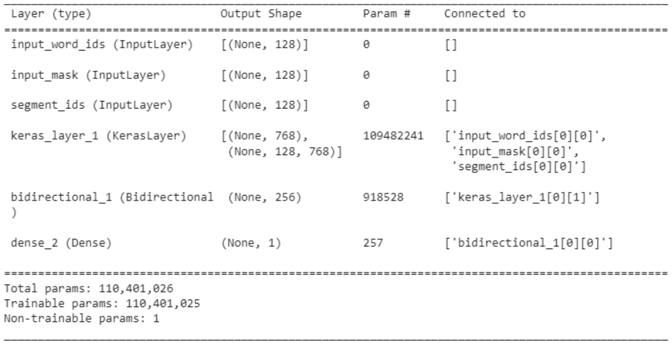
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_11 (InputLayer)       [(None, 224, 224, 3)]     0

 vgg16 (Functional)          (None, 1000)              138357544

 flatten (Flatten)           (None, 1000)              0

 dense_13 (Dense)            (None, 512)               512512

 dropout_2 (Dropout)         (None, 512)               0

 dense_14 (Dense)            (None, 1)                 513

=================================================================
Total params: 138,870,569
Trainable params: 513,025
Non-trainable params: 138,357,544
_________________________________________________________________
```

Fig 3: Structure of Model 2 (Base VGG-16 with relu activation and dropout)

5. TEXT MODEL

To achieve the best value of accuracy in the text model, we compared 3 different text models with sigmoid activation in the last layer for binary classification, Model 1 is a base pre-trained Model BERT with GlobalMaxPool1D, Model 2 is a base pre-trained model BERT combined with one layer of LSTM, Model 3 is a base pre-trained Model BERT with the combination of CNN and LSTM. The parameter of neural is 128, the early stopping monitor on validation loss with patience equal to 2, and for the learning strategy, we used exponential scheduling to make sure the learning rate will decrease as the loss function converges. For the model compilation, we selected binary cross entropy to loss, optimizer as Adam, and metrics with accuracy.

The evaluation accuracy of all the models on testing sets is 0.5876. The plot below is the structure of Model 2.



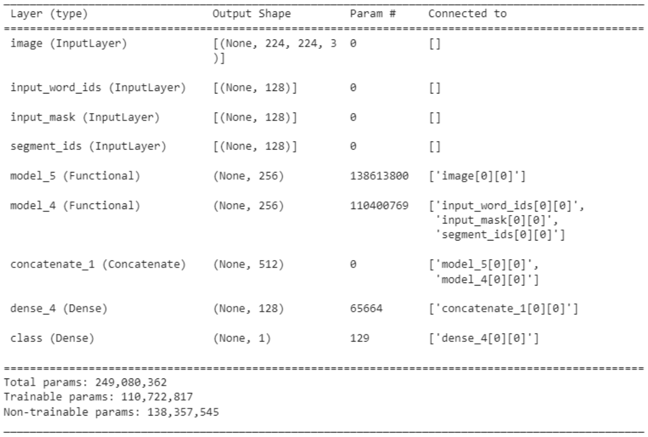
```
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_word_ids (InputLayer)    [(None, 128)]        0           []

 input_mask (InputLayer)        [(None, 128)]        0           []

 segment_ids (InputLayer)       [(None, 128)]        0           []

 keras_layer_1 (KerasLayer)     [(None, 768),        109482241   ['input_word_ids[0][0]',
                                 (None, 128, 768)]                'input_mask[0][0]',
                                                                  'segment_ids[0][0]']

 bidirectional_1 (Bidirectional  (None, 256)         918528      ['keras_layer_1[0][1]']
 )

 dense_2 (Dense)                (None, 1)            257         ['bidirectional_1[0][0]']

==================================================================================================
Total params: 110,401,026
Trainable params: 110,401,025
Non-trainable params: 1
__________________________________________________________________________________________________
```

Fig 4: Structure of Model 2 (Base pre-trained model BERT combined with one layer of LSTM)

6. EARLY FUSION MODEL

For the early fusion, we fusion the base pre-train model with VGG-16 and BERT. The parameter of the output layer neural is 256, the early stopping monitor on validation loss with patience equal to 5, and for the learning strategy, we used exponential scheduling to make sure the learning rate will decrease as the loss function converges. For the model compilation, we selected binary cross entropy to loss, and optimizer as Adam.

The evaluation accuracy of the model on testing sets is 0.9375. Fig 5 indicates the structure of the early fusion model, and Fig 6 shows the coverage status versus learning rate in the early fusion model.



```
____________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
============================================================================================
 image (InputLayer)             [(None, 224, 224, 3  0           []
                                )]

 input_word_ids (InputLayer)    [(None, 128)]        0           []

 input_mask (InputLayer)        [(None, 128)]        0           []

 segment_ids (InputLayer)       [(None, 128)]        0           []

 model_5 (Functional)           (None, 256)          138613800   ['image[0][0]']

 model_4 (Functional)           (None, 256)          110400769   ['input_word_ids[0][0]',
                                                                  'input_mask[0][0]',
                                                                  'segment_ids[0][0]']

 concatenate_1 (Concatenate)    (None, 512)          0           ['model_5[0][0]',
                                                                  'model_4[0][0]']

 dense_4 (Dense)                (None, 128)          65664       ['concatenate_1[0][0]']

 class (Dense)                  (None, 1)            129         ['dense_4[0][0]']

============================================================================================
Total params: 249,080,362
Trainable params: 110,722,817
Non-trainable params: 138,357,545
____________________________________________________________________________________________
```

Fig 5: Structure of early fusion model

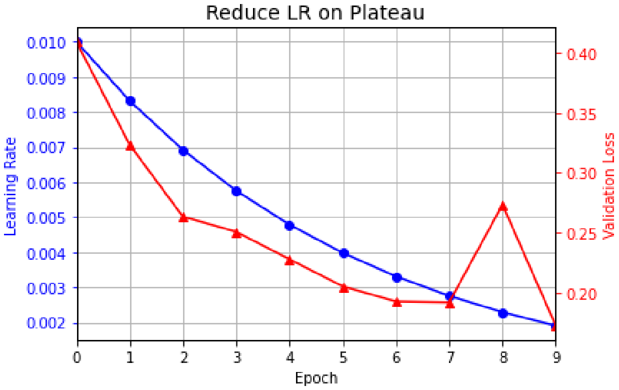
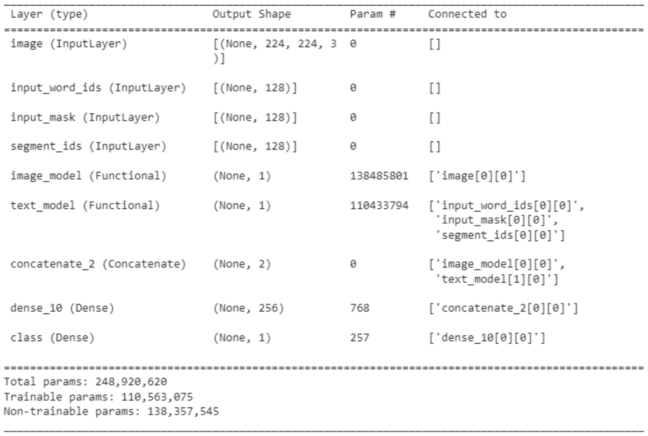


Fig 6: Learning rate vs validation loss (Early fusion)

7. LATE FUSION MODEL

For the late fusion, we combined the base pre-train model with VGG-16 and BERT. The parameter of the output layer neural is 1 with sigmoid activation, the early stopping monitor on validation loss with patience equal to 5, and for the learning strategy, we used exponential scheduling to make sure the learning rate will decrease as the loss function converges. For the model compilation, we selected binary cross entropy to loss, and optimizer as Adam.

The evaluation accuracy of the model on testing sets is 0.95. Fig 7 indicates the structure of the late fusion model, and Fig 8 shows the coverage status versus learning rate in the late fusion model.



```
____________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
============================================================================================
 image (InputLayer)             [(None, 224, 224, 3  0           []
                                )]

 input_word_ids (InputLayer)    [(None, 128)]        0           []

 input_mask (InputLayer)        [(None, 128)]        0           []

 segment_ids (InputLayer)       [(None, 128)]        0           []

 image_model (Functional)       (None, 1)            138485801   ['image[0][0]']

 text_model (Functional)        (None, 1)            110433794   ['input_word_ids[0][0]',
                                                                  'input_mask[0][0]',
                                                                  'segment_ids[0][0]']

 concatenate_2 (Concatenate)    (None, 2)            0           ['image_model[0][0]',
                                                                  'text_model[1][0]']

 dense_10 (Dense)               (None, 256)          768         ['concatenate_2[0][0]']

 class (Dense)                  (None, 1)            257         ['dense_10[0][0]']

============================================================================================
Total params: 248,920,620
Trainable params: 110,563,075
Non-trainable params: 138,357,545
____________________________________________________________________________________________
```

Fig 7: Structure of early fusion model

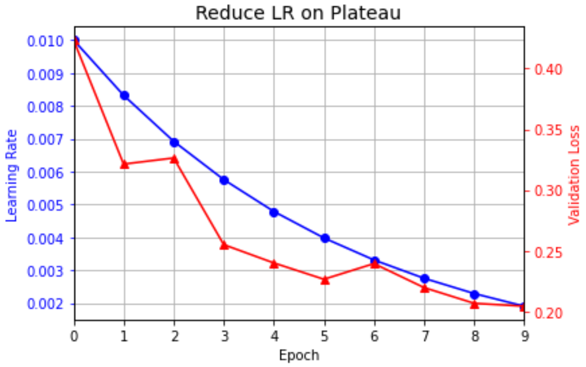


Fig 8: Learning rate vs validation loss (Late fusion)

8. CONCLUSION AND FUTURE WORKS

According to the performance of the fusion model, we success to improve the accuracy of the original model. For the image model, the highest accuracy that we obtained is 0.72, and 0.58 for the text model. In addition, the performance of early fusion has higher accuracy than late fusion, and the fusion model is indeed increasing the accuracy of the single model. The reason why we did not choose the mode with the highest accuracy is that we found the accuracy decreased slightly in the late fusion model if we use model 2 in the image model (0.92). For the future direction, we are interested in incorporating additional modalities in our study, since various models can also increase the variety of the model. Moreover, to increase the performance of misinformation detection, we can enhance the number of modalities from the multivariate data fusion model.

9. REFERENCES

[1] Damasceno, L. P., Shafer, A., Japkowicz, N., Cavalcante, C. C., & Boukouvalas, Z. (2022). Efficient multivariate data fusion for misinformation detection during high impact events. *Discovery Science*, 253–268. https://doi.org/10.1007/978-3-031-18840-4_19

[2] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. https://doi.org/10.48550/arXiv.1810.04805

[3] Rendy, k. (2022, December 2). Tuning the hyperparameters and layers of neural network deep learning. Analytics Vidhya. Retrieved December 4, 2022, from https://www.analyticsvidhya.com/blog/2021/05/tuning-the-hyperparameters-and-layers-of-neural-network-deep-learning/

[4] Ruby, D. (2022, November 30). Twitter statistics: Facts and figures after Elon Musk takeover (2022). demandsage. Retrieved December 2, 2022, from https://www.demandsage.com/twitter-statistics/

[5] Stewart, M. (2020, July 29). Simple guide to hyperparameter tuning in Neural Networks. Medium. Retrieved December 4, 2022, from https://towardsdatascience.com/simple-guide-to-hyperparameter-tuning-in-neural-networks-3fe03dad8594

[6] Thakur, R. (2020, November 24). Step by step VGG16 implementation in Keras for beginners. Medium. Retrieved December 3, 2022, from https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c