MULTIMODAL DEEP LEARNING

Sihyuan Han

ABSTRACT

Nowadays, Memes are being wildly used on the Internet, especially on social media. People use memes to reflect on current events, criticize trending news, and so on. Unlike comment and GIF, meme generates text and image and becomes a multimodal type of speech. The dataset "ImgFlip-Scraped Memes Caption" has been used in this project. There are over 200 thousand samples with 81 different labels in this dataset, each label has over a thousand rows. Columns that are being used in this project include text extracted from Memes (named CaptionText), memes (stored in image URL), and labels. The proposed research consists of three parts, text analysis, image analysis, and multimodal (text and image) analysis. By building supervised classification models, the result accuracy performance of each model would be compared.

keywords: Memes, Multimodal, Model, Neural network, Deep learning, Transfer Learning

I. INTRODUCTION

"Meme" is a concept of humor spread by people on the internet. It has been around since the beginning of the internet and was made massively popular when social media popped up. At first, memes have been based on a certain category such as "grumpy cat" which was popular in the early 2010s. In the past two decades, memes have become one of the primary forms of digital communication. They are passionately used by people for different purposes. For example, people's self-expression of their emotions, advertisements for businesses to gain exposure, and political and other religious reasons can also be seen on most social media platforms. Based on numerous types of memes, it would be interesting to detect different classes of memes. However, memes analysis could be difficult due to the multimodal kind of feature (text + image). Thus, the technique used in this study would be merging both text and image models to form a new multi-model.

In the following, the project had been divided into three parts. First, a text classification model, using the pre-trained model "BERT". Next, an image binary classification model using the pre-trained model "VGG-16". Last but not least, is the fusion model which is a multimodal model that is merged by text and image, and both early fusion and late fusion will be presented in this paper and compare which model will have a better outcome, that is a better accuracy of identifying the label.

II. DATA PREPROCESSING

The "ImgFlip-Scraped Memes Caption" dataset contains 5 variables. Duplicated data has been removed based on hash id. In order to successfully access the image link for every picture in the dataset, "HTTP" has been added to all the images, and images that turn out to be forbidden to access have been removed. Text variables in this dataset have been converted to string type instead of an object.

For effective training, the dataset would be subset to two-class and five-class to conduct binary and multiclass supervised classification. The labels that have been chosen for binary classification include "Who Killed Hannibal" and "Scared Cat". For multiclass classification, "Sleeping Shaq", "Uncle Sam", and "Peter Parker Cry" are being added to the data. When splitting the data into training and testing, the test set is split into 25% of actual data, both in binary and multiclass classification models.

Two data pre-processing approaches are being used in these models, that is traditional train-test-split and data generator. Both methods have its advantage in different models. For the text model in the study, it is easier to use the traditional train-test-split because the generator with BERT pre-trained model requires writing a function to iterate the tokens and labels. Traditional methods reduce the time it may spend on writing and defining the type of each token and the pre-trained model's required input shape. For the image model and fusion model, it is more efficient to use the data generator since the image requires a lot of memory when feeding into the model, the iteration feature of the generator simply solves this problem and also makes the code cleaner and more understandable.

II. METHODOLOGY

A. Image Model

VGG-16 is being used as the base of the image pre-trained model in this study. VGG-16 is a CNN (Convolutional Neural Network) model for image recognition proposed by the University of Oxford, VGG-16 refers to a VGG model with 16 weight layers, there is also a VGG-19 model which is with 19 weight layers. It uses a 3x3 layer instead of 5x5 or 7x7 to do feature extraction for better accuracy. The pooling layer is 2x2 while other pre-trained models such as AlexNet is 3x3. It requires image input to be (224, 224, 3). Below is a basic structure of the CNN model.

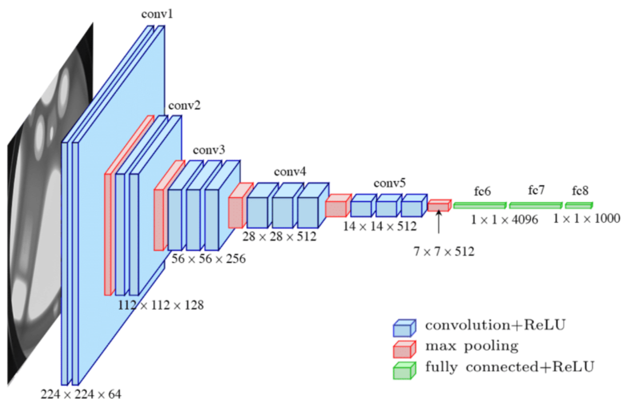


Fig.1. CNN (Convolutional Neural Network) Structure

B. Text Model

For text models, BERT is being used as the pre-trained model. BERT, which stands for Bidirectional Encoder Representations from Transformers, is Published in 2018 by Google, just like its name, it learns the text representation from both directions instead of only from left to right or right to left, so it can get a better sense of the context and its relationship. BERT consists of many different pre-trained models such as "base", "large", "uncased" and so on, with different languages including English, Chinese, and over a hundred languages. The Structure of the BERT model is presented like this: a single Token that starts with [CLS] which stands for "classification", then follows with the input sentence, then ends up with [SEP] which tells the machine to separate each sentence. In this project, BERT "Base" is chosen to be used as the pre-trained model. The architecture of the "Base" model is that there are 12 transformer blocks, 768 hidden layers, and 12 attention heads. BERT requires specifically formatted inputs, for each tokenized input sentence, it needs "Input Id", "Segment Mask", "Attention Mask" and "Label". "Input Id" is a sequence of integers identifying each input token to its index number in the BERT tokenizer vocabulary. "Segment Mask" is a sequence of 1s and 0s used to identify whether the input is one sentence or two sentences long. "Attention Mask" is a sequence of 1s and 0s as well, 1s for all input tokens and 0s for all padding tokens. For example, if max_length is set to 128, the input sentence less than 128 will be padded with 0 to 128. "Label" is presented as a single numeric value like 0 or 1 (binary classification).

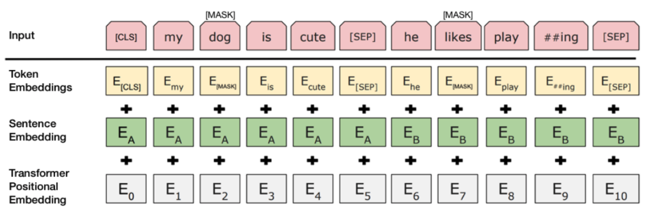


Fig.2. BERT Structure, Source: BERT [Devlin et al., 2018]

C. Multimodal Model

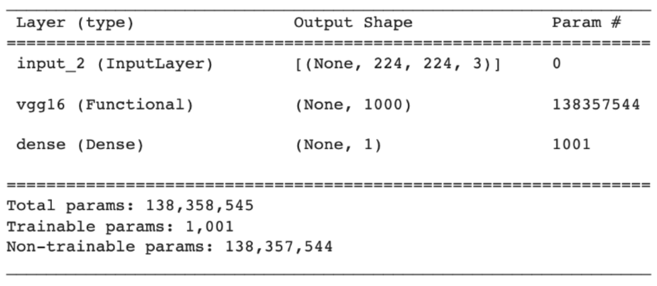
Diving into multimodal deep learning, by utilizing text and image features, the project is going to perform multimodal classification analysis using early fusion and late fusion. The difference between early and late fusion is that early fusion starts at the feature level, which means many features are concatenated into one super-vector for further classification, and late fusion fuses the modalities in the classification scores level. Late fusion is commonly used when the data sources are significantly varied from each other. For example, data with major differences in sampling rate, data dimensionality, and unit of measurement.

In order to accelerate the training process, transfer learning is implemented in both fusion methods. Image features are extracted by applying the VGG-16 model as the pre-trained model, while text features are extracted by adopting BERT as a pre-trained model to learn vector representations of words. A bidirectional LSTM layer is also being added to fine-tune the text model in the fusion state.

III. EXPERIMENT AND RESULTS

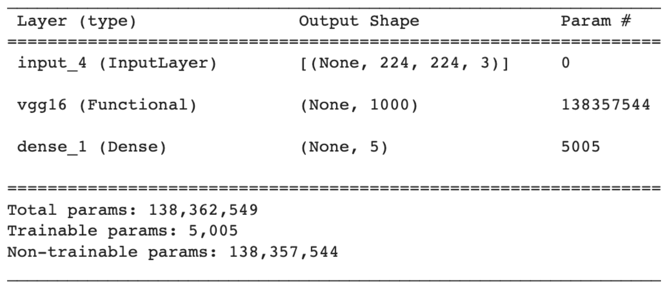
A. Image Model

The activation functions used in the image model are "sigmoid" for binary class and "softmax" for multiclass. Setting "Adam" as the optimizer with a learning rate of 0.005. While compiling the model, the loss functions that are chosen to be used are "binary_crossentropy" and "categorical_crossentropy". The following is the summary of binary and multiclass models.



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 224, 224, 3)]     0

 vgg16 (Functional)          (None, 1000)              138357544

 dense (Dense)               (None, 1)                 1001

=================================================================
Total params: 138,358,545
Trainable params: 1,001
Non-trainable params: 138,357,544
_________________________________________________________________
```

Fig.3. Binary classification image model applied VGG-16



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_4 (InputLayer)        [(None, 224, 224, 3)]     0

 vgg16 (Functional)          (None, 1000)              138357544

 dense_1 (Dense)             (None, 5)                 5005

=================================================================
Total params: 138,362,549
Trainable params: 5,005
Non-trainable params: 138,357,544
_________________________________________________________________
```

Fig.4. Multiclass classification image model applied VGG-16

The difference between the two models is the output dense unit, binary mode is one 1 and multiclass is n class, so in this project equals to 5.

The binary classification model is expected to have higher accuracy than the binary classification model since the more class in the model, the harder for machines to learn and classify. During training in these models, both models have been trained in 10 epochs. Below is a chart comparing the two models' accuracy.

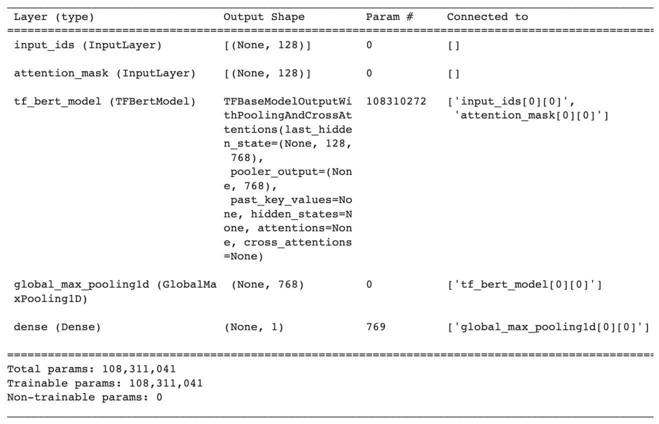
	binary	multiclass
epoch 1 accuracy	0.93	0.32
epoch 10 accuracy	**0.99**	**0.99**
accuracy > 0.5	epoch 1	epoch 2

Fig.5. Accuracy comparison of image models

Based on the table, the binary model's accuracy already has 0.93 in epoch 1 while the multiclass model has only 0.32. Reaching epoch 10, both models have an accuracy of 0.99. The table also added the row that tells the reader which epoch when the training accuracy gets to over 0.5.

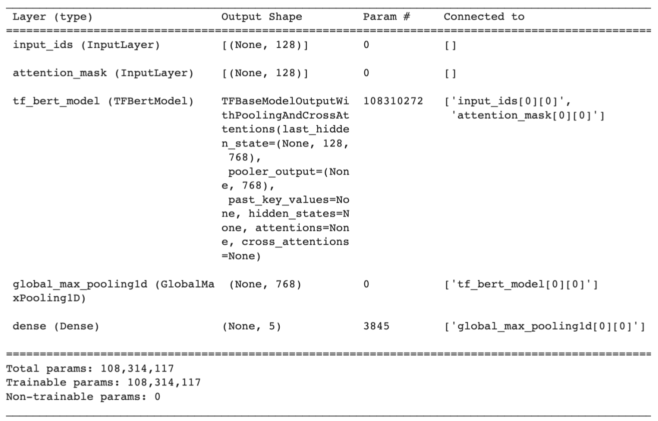
B. Text Model

The activation used in the text model is "sigmoid" for binary class and "softmax" for multiclass, as same as the settings in the image models. . The difference is that for text models, the learning rate is being set as default to 3e-5 with "Adam" optimizer. The loss functions used when compiling the models are the same as the image model as well. The following is the summary of binary and multiclass text models.



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

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

tf_bert_model (TFBertModel)    TFBaseModelOutputWi  108310272   ['input_ids[0][0]',
                               thPoolingAndCrossAt               'attention_mask[0][0]']
                               tentions(last_hidde
                               n_state=(None, 128,
                                768),
                                pooler_output=(Non
                               e, 768),
                                past_key_values=No
                               ne, hidden_states=N
                               one, attentions=Non
                               e, cross_attentions
                               =None)

global_max_pooling1d (GlobalMa  (None, 768)         0           ['tf_bert_model[0][0]']
xPooling1D)

dense (Dense)                  (None, 1)            769         ['global_max_pooling1d[0][0]']

====================================================================================
Total params: 108,311,041
Trainable params: 108,311,041
Non-trainable params: 0
____________________________________________________________________________________
```

Fig.6. Binary classification text model adopted BERT



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

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

tf_bert_model (TFBertModel)    TFBaseModelOutputWi  108310272   ['input_ids[0][0]',
                               thPoolingAndCrossAt               'attention_mask[0][0]']
                               tentions(last_hidde
                               n_state=(None, 128,
                                768),
                                pooler_output=(Non
                               e, 768),
                                past_key_values=No
                               ne, hidden_states=N
                               one, attentions=Non
                               e, cross_attentions
                               =None)

global_max_pooling1d (GlobalMa  (None, 768)         0           ['tf_bert_model[0][0]']
xPooling1D)

dense (Dense)                  (None, 5)            3845        ['global_max_pooling1d[0][0]']

====================================================================================
Total params: 108,314,117
Trainable params: 108,314,117
Non-trainable params: 0
____________________________________________________________________________________
```

Fig.7. Multiclass classification text model adopted BERT

The difference between the two models is also the output dense unit, binary mode is one 1, and multiclass is 5. Below is a graph comparing the two text models' accuracy.

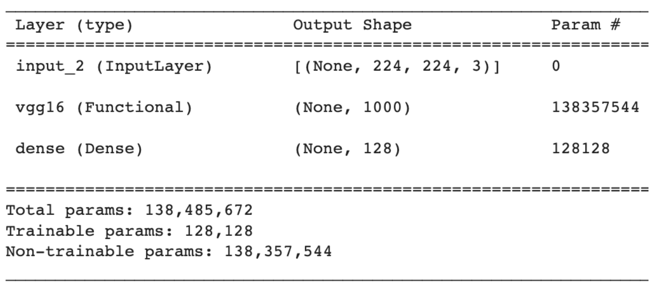
	binary	multiclass
epoch 1 accuracy	0.86	0.82
epoch 10 accuracy	**0.99**	**0.98**
accuracy > 0.5	epoch 1	epoch 1

Fig.8. Accuracy comparison of text models

Learning from the table, from epoch 1 to epoch 10, both models have similar accuracy, starting from the first epoch, both models have an accuracy of more than 0.8. In the end, both models reach to an accuracy of 0.99.

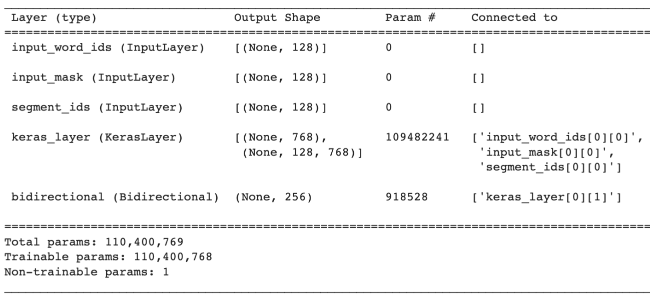
C. Multimodal Model

After building the image and text-independent models, the questions are whether the fusion models have better accuracy in classifying the labels, and between early fusion and late fusion which has the higher accuracy of predicting the labels? In the following section, only multiclass models will be shown as examples of the models' construction.



```
Layer (type)                Output Shape              Param #
=================================================================
input_2 (InputLayer)        [(None, 224, 224, 3)]     0

vgg16 (Functional)          (None, 1000)              138357544

dense (Dense)               (None, 128)               128128

=================================================================
Total params: 138,485,672
Trainable params: 128,128
Non-trainable params: 138,357,544
_________________________________________________________________
```

Fig.9. Multiclass classification image model before early fusion



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

 bidirectional (Bidirectional)  (None, 256)          918528      ['keras_layer[0][1]']

==================================================================================================
Total params: 110,400,769
Trainable params: 110,400,768
Non-trainable params: 1
```

Fig.10. Multiclass classification text model before early fusion

As the pictures are shown above, the output dense unit of the image model is 128, which is because the text model's input max_length is set to 128. The output dense unit of the text model is 256 because a bidirectional LSTM layer is added to fine-tune the model. Here is a graph that presents the early fusion model after concatenation.

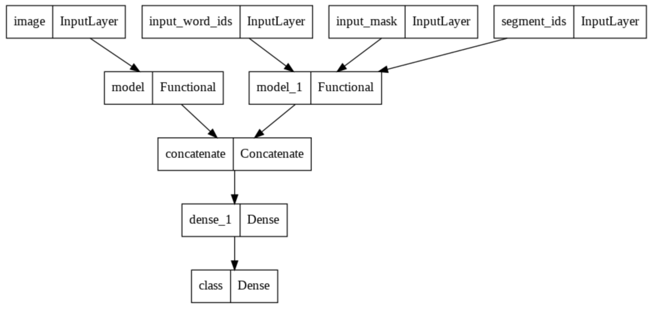
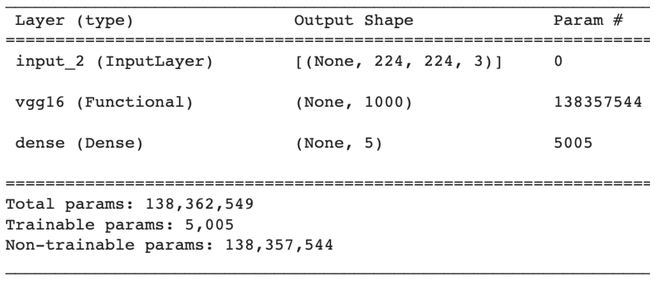


Fig.11. Multiclass classification early fusion model after concatenation

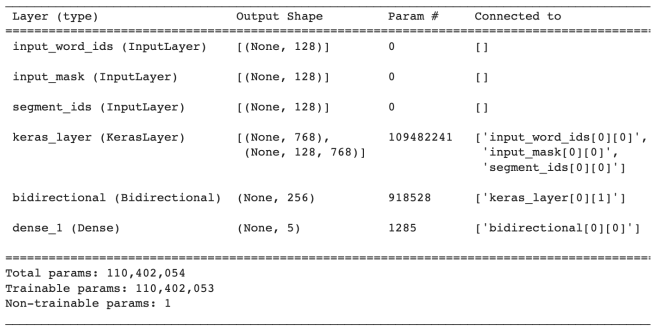
The final output dense unit is 5 (multiclass model), and the model compiles with the same settings as the independent image and text model. Compile with "Adam" optimizer with default learning rate 3e-5.

Jump into the late fusion, below are the models' summaries before merging into the late fusion multi-model.



```
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 224, 224, 3)]     0

 vgg16 (Functional)          (None, 1000)              138357544

 dense (Dense)               (None, 5)                 5005

=================================================================
Total params: 138,362,549
Trainable params: 5,005
Non-trainable params: 138,357,544
```

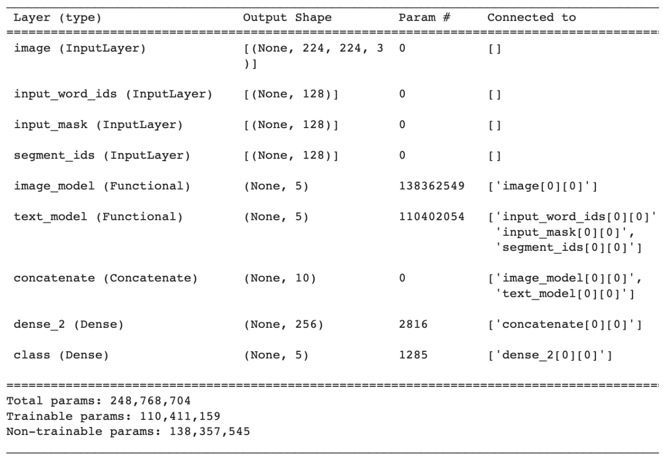
Fig.12. Multiclass classification image model before late fusion



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

 bidirectional (Bidirectional)  (None, 256)          918528      ['keras_layer[0][1]']

 dense_1 (Dense)                (None, 5)            1285        ['bidirectional[0][0]']

==================================================================================================
Total params: 110,402,054
Trainable params: 110,402,053
Non-trainable params: 1
```

Fig.13. Multiclass classification text model before late fusion

Late fusion joins the two independent models and then combines the prediction scores of multiple classifiers. It could be seen that instead of 128 and 256 dense units of models using the early fusion method, the output dense units of text and image model used for late fusion before merging are both 5. It is also shown in the merged late fusion model that the last layer of the output dense unit of the late fusion model is also 5, that is because the multiclass model has 5 different labels. Here is the summary of 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, 5)            138362549   ['image[0][0]']

 text_model (Functional)        (None, 5)            110402054   ['input_word_ids[0][0]'
                                                                  'input_mask[0][0]',
                                                                  'segment_ids[0][0]']

 concatenate (Concatenate)      (None, 10)           0           ['image_model[0][0]',
                                                                  'text_model[0][0]']

 dense_2 (Dense)                (None, 256)          2816        ['concatenate[0][0]']

 class (Dense)                  (None, 5)            1285        ['dense_2[0][0]']

==================================================================================================
Total params: 248,768,704
Trainable params: 110,411,159
Non-trainable params: 138,357,545
```

Fig.14. Multiclass classification late fusion model after merging

The final output dense unit is also 5 (multiclass model), the model compiles with the same settings as the early fusion model.

To thoroughly compare the accuracy between early and late fusion model, two tables and plots will be presented in the sequence of binary models with early and late fusion and then multiclass models with early and late fusion. The following is the table of binary classification models accuracy comparison from epoch 1 to 10.

	early fusion	late fusion
epoch 1 accuracy	0.53	0.51
epoch 10 accuracy	**0.99**	**0.99**
accuracy > 0.5	epoch 1	epoch 1

Fig.15. Accuracy comparison table of the binary fusion model

In the binary classification table with early and late fusion, the result has no big difference between both models. These models have around 0.5 accuracies in the first epoch, then both reach 0.99 in epoch 10. Every epoch's training and validation with accuracy and loss are shown in the below plots.

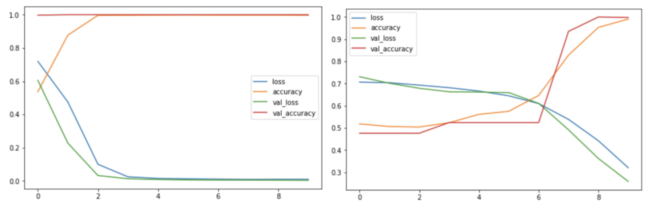


Fig.16. Left: early fusion result of the binary classification model
Right: late fusion result of the binary classification model

The possible reason for this outcome is that there are only two classes, it's easier for machines to distinguish the labels of the memes.

The upcoming table is multiclass classification models with early and late fusion accuracy comparison from epoch 1 to 10.

	early fusion	late fusion
epoch 1 accuracy	0.38	0.22
epoch 10 accuracy	**0.99**	**0.79**
accuracy > 0.5	epoch 2	epoch 8

Fig.17. Accuracy comparison table of the multiclass fusion model

In the multiclass classification table with early and late fusion, the result is significantly different than binary models. In the early fusion model, the first epoch has only 0.38 accuracy, but it went over 0.5 at epoch 2, and in the end, it reaches 0.99 as in other models. In contrast, for the late fusion model, the first epoch has only 0.22 accuracy, and it takes 8 epochs to reach an accuracy over 0.5, eventually, the model ends with an accuracy of 0.79. Every epoch's training and validation with accuracy and loss are shown in the below plots.

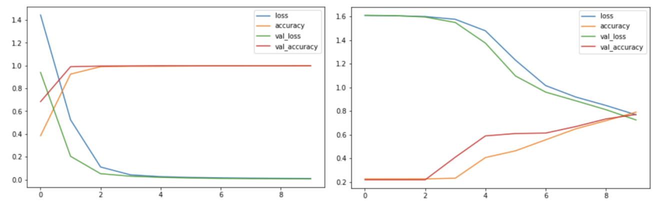


Fig.18. Left: early fusion result of the multiclass classification model
Right: late fusion result of the multiclass classification model

The cause of the result could be the flexibility of the late fusion model because late fusion operates on inferences and not feature levels, it is not effective at modeling signal-level interactions between modalities. It uses data sources independently, which may ignore some lower-level interactions between image and text features.

IV. OBSTACLES

A few problems have occurred in the study. First, while tidying up the image before image preprocessing, some images' URLs showed 403 forbidden when tried to download them to the local file, so those images ended up being removed in the final dataset. Second, at the beginning of training the image model, the accuracy is always staying around 0.5. After fine-tuning and hyper-parameter tuning the model, the reason has been found out that the activation function used in the binary model should be "sigmoid" instead of "softmax". Last but not least, basic train-test-split (x_train, x_test, y_train, y_test) and data generator can both be seen used in the code, that is because when starting to load the data, the generator has been used in all the models. However, defining a function for special tokens that are used in the BERT model takes a big amount of time. Thus, traditional train-test-split to label and features reduces the time of writing complicated code. It is also a good opportunity to pick the most efficient method when building many models.

V. CONCLUSION AND FUTURE WORK

To sum up, there are several findings that can be concluded from the study. First, early fusion has better accuracy than late fusion in detecting labels for this dataset. Second, binary models have higher accuracy than multiclass models. The reason is obvious that it is easier for machines to determine with the lower class of labels. Last, fusion models' accuracy is not clearly higher than the independent image or text models. The scenario might be the similarity of memes images in this dataset, or because of the limitation of training 5 classes instead of more (the original dataset has 81 classes). In future steps, researchers could run more classes on multiclass classification or pick similar classes to do prediction. For example, humans, animals, or anime.

VI. REFERENCES

[1] Wang, Z., Yin, Z., &amp; Argyris, Y. A. (2021). Detecting medical misinformation on social media using Multimodal Deep Learning. IEEE Journal of Biomedical and Health Informatics, 25(6), 2193–2203. https://doi.org/10.1109/jbhi.2020.3037027

[2] Aggarwal, P., Liman, M. E., Gold, D., &amp; Zesch, T. (2021). VL-Bert+: Detecting protected groups in hateful multimodal memes. Proceedings of the 5th Workshop on Online Abuse and Harms(WOAH 2021). https://doi.org/10.18653/v1/2021.woah-1.22

[3] Chiu, C. Y., Lane, H. Y., Koh, J. L., &amp; Chen, A. L. (2020). Multimodal depression detection on Instagram considering time interval of posts. Journal of Intelligent Information Systems, 56(1), 25–47. https://doi.org/10.1007/s10844-020-00599-5

[4] Hu, J., Yamasaki, T., &amp; Aizawa, K. (2016). Multimodal learning for image popularity prediction on social media. 2016 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW). https://doi.org/10.1109/icce-tw.2016.7521017

[5] Chandra, M., Pailla, D., Bhatia, H., Sanchawala, A., Gupta, M., Shrivastava, M., &amp; Kumaraguru, P. (2021). “subverting the jewtocracy”: Online antisemitism detection using multimodal deep learning. 13th ACM Web Science Conference 2021. https://doi.org/10.1145/3447535.3462502

[6] Dong, Y., Gao, S., Tao, K., Liu, J., &amp; Wang, H. (2013). Performance evaluation of early and late fusion methods for generic semantics indexing. Pattern Analysis and Applications, 17(1), 37–50. https://doi.org/10.1007/s10044-013-0336-8

[7] Badour, J., &amp; Brown, J. A. (2021). Hateful memes classification using machine learning. 2021 IEEE Symposium Series on Computational Intelligence (SSCI). https://doi.org/10.1109/ssci50451.2021.9659896

[8] Abousaleh, F., Cheng, W., Yu, N., &amp; Tsao, Y. (2020). Multimodal Deep Learning Framework for Image Popularity Prediction on Social Media. https://doi.org/10.1109/TCDS.2020.3036690