DL A4

DRISHYA UNIYAL	**MT21119**
PRASHANT SHARMA	**MT21227**

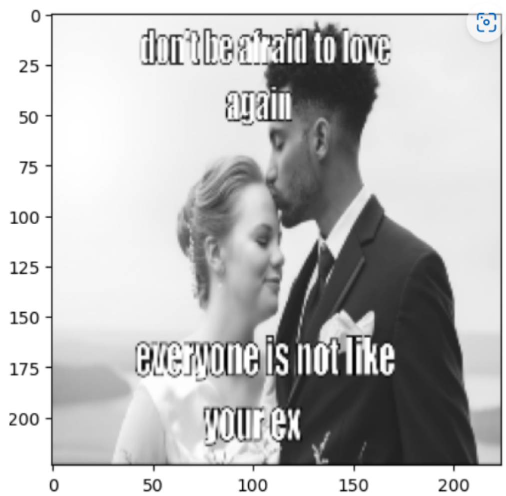
DATASET for TASK I, II & III

Dataset description: Dataset of hateful and not hateful memes

● **Familiarize yourself with the dataset and understand of what makes a meme hateful. Perform the necessary data cleaning/ preprocessing steps. In case some image is missing corresponding to jsonl, ignore it.**

Steps:

1. Get the train, test dev sets in a dataframe.
2. Load image using cv2
3. Resize image to 224x224
4. Convert image to grayscale
5. Normalize pixel values to [0, 1] range
6. Add channel dimension to image



● Use any Deep Learning libraries like Pytorch or TF to develop deep learning model.
● DO NOT CHANGE THE TEST SAMPLES SIZE FOR PERFORMANCE COMPARISION.
● If you reduce the number of train for computational purpose, it should be stratitifed proportionally, and stated clearly in the report.

TASK I (Unimodal: Image-Only) [20 marks]

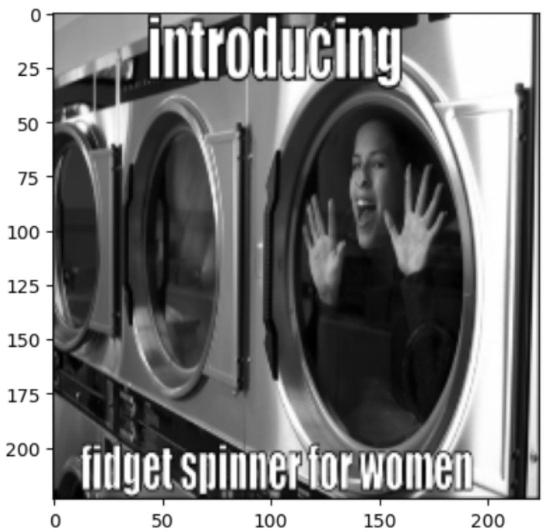
Image-only hateful meme detection

Task: Develop image-only detection system to classify memes as hateful or not hateful.
Consider image as a whole and do not treat text as separate component in it.
Use train/val/test sets provided in jsonl format in link shared above.

1. **Pre-process the images using atleast 2 techniques of your choice as converting into appropriate format, normalization, gray-scaling etc. Make sure that your images are in the appropriate format for your chosen model. [2 marks]**

Pre - processing:

1. Load the image
2. Resize image to 224x224
3. Convert image to grayscale
4. Normalize pixel values to [0, 1] range
5. Add channel dimension to image
6. Convert them to arrays.



2. **Propose and implement an image-only model for classification using any deep learning imag classification model of your choice such as VGG, Vision Transformer Image-only, ResNet, etc or you can build your own CNN model. The model selected should learn meaningful features from images and be effective for image-only classification tasks. [12 marks]**

Sol:

We have implemented simple **CNN model** with variations and tested for the best model.

Model 1:

1. Simple model with input shape (224,224,1)
2. Conv2D with Max Pooling layers.
3. Loss = binary_crossentropy
4. Optimizer = Adam (lr = 0.001)
5. Activation : relu and sigmoid(for output)

Model 2:

1. Conv2D with Max Pooling layers.

2. Batch Normalization layer.
3. Dropout with 0.25
4. Loss = binary_crossentropy
5. optimizer = Adam (lr = 0.001)
6. Activation: relu and sigmoid(for output)

Model 3:

1. Conv2D with Amx Pooling layers
2. Dropout with 0.5
3. Loss = binary_crossentropy
4. optimizer = Adam (lr = 0.001)
5. Activation: relu and sigmoid(for output)

-The models have been trained then for **50 epochs**.
- Different functions for precision, recall, and f1 have been made.
- function for plotting graphs has been made.
- function for getting all the metrics have been made.

Model 1 Results:

```
Following are the TEST metrics associated with the model

Test loss: 3.4917807579040527

Test acuuracy: 0.4950000047683716

Test precision: 0.4740484356880188

Test recall: 0.279591828584671

Test F1-Score: 0.34587135910987854

32/32 [==============================] - 0s 4ms/step
              precision    recall  f1-score   support

           0       0.50      0.70      0.59       510
           1       0.47      0.28      0.35       490

    accuracy                           0.49      1000
   macro avg       0.49      0.49      0.47      1000
weighted avg       0.49      0.49      0.47      1000
```

```
Following are the TRAIN metrics associated with the model

Train loss: 0.018920116126537323

Train acuuracy: 0.9956470727920532

Train precision: 0.9933818578720093

Train recall: 0.9943689703941345

Train F1-Score: 0.9932110905647278

266/266 [==============================] - 1s 4ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5481
           1       0.99      0.99      0.99      3019

    accuracy                           1.00      8500
   macro avg       1.00      1.00      1.00      8500
weighted avg       1.00      1.00      1.00      8500
```

(This is for the Dev metrics by mistake printing statement says Train.)

```
Following are the DEV metrics associated with the model

Train loss: 3.645599126815796

Train acuuracy: 0.4779999852180481

Train precision: 0.4485294222831726

Train recall: 0.24696356058120728

Train F1-Score: 0.3075941503047943

16/16 [==============================] - 0s 4ms/step
              precision    recall  f1-score   support

           0       0.49      0.70      0.58       253
           1       0.45      0.25      0.32       247

    accuracy                           0.48       500
   macro avg       0.47      0.48      0.45       500
weighted avg       0.47      0.48      0.45       500
```

Model 2 Results::

```
Test loss: 0.7309284210205078

Test acuuracy: 0.5099999904632568

Test precision: 0.0

Test recall: 0.0

Test F1-Score: 0.0

32/32 [==============================] - 0s 10ms/step
              precision    recall  f1-score   support

           0       0.51      1.00      0.68       510
           1       0.00      0.00      0.00       490

    accuracy                           0.51      1000
   macro avg       0.26      0.50      0.34      1000
weighted avg       0.26      0.51      0.34      1000
```

Following are the TRAIN metrics associated with the model

Train loss: 0.6505924463272095

Train acuuracy: 0.6448235511779785

Train precision: 0.0

Train recall: 0.0

Train F1-Score: 0.0

266/266 [==============================] - 3s 10ms/step

	precision	recall	f1-score	support
0	0.64	1.00	0.78	5481
1	0.00	0.00	0.00	3019
accuracy			0.64	8500
macro avg	0.32	0.50	0.39	8500
weighted avg	0.42	0.64	0.51	8500

Following are the DEV metrics associated with the model

Train loss: 0.7333117723464966

Train acuuracy: 0.5059999823570251

Train precision: 0.0

Train recall: 0.0

Train F1-Score: 0.0

	precision	recall	f1-score	support
0	0.51	1.00	0.67	253
1	0.00	0.00	0.00	247
accuracy			0.51	500
macro avg	0.25	0.50	0.34	500
weighted avg	0.26	0.51	0.34	500

Model 3 Results:

Following are the TEST metrics associated with the model

Test loss: 4.72576904296875

Test acuuracy: 0.5170000195503235

Test precision: 0.5140562057495117

Test recall: 0.2612244784832001

Test F1-Score: 0.3352556824684143

32/32 [==============================] - 0s 5ms/step

	precision	recall	f1-score	support
0	0.52	0.76	0.62	510
1	0.51	0.26	0.35	490
accuracy			0.52	1000
macro avg	0.52	0.51	0.48	1000
weighted avg	0.52	0.52	0.48	1000

Following are the TRAIN metrics associated with the model

Train loss: 6.557403685292229e-05

Train acuuracy: 1.0

Train precision: 1.0

Train recall: 1.0

Train F1-Score: 1.0

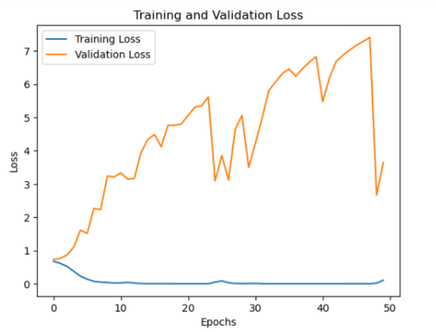
266/266 [==============================] - 1s 5ms/step

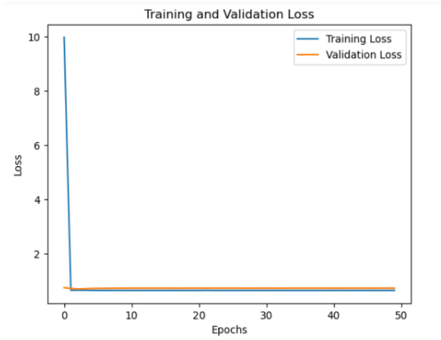
	precision	recall	f1-score	support
0	1.00	1.00	1.00	5481
1	1.00	1.00	1.00	3019
accuracy			1.00	8500
macro avg	1.00	1.00	1.00	8500
weighted avg	1.00	1.00	1.00	8500

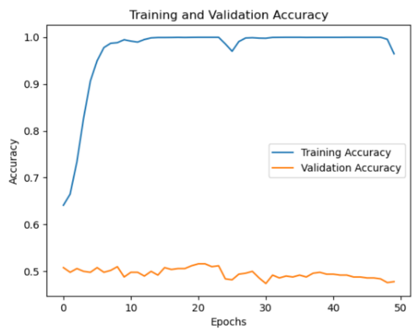
```
Following are the DEV metrics associated with the model

Train loss: 5.097967624664307

Train acuuracy: 0.48399999737739563

Train precision: 0.45045045018196106

Train recall: 0.2024291455745697

Train F1-Score: 0.24692636728286743

16/16 [==============================] - 0s 5ms/step
              precision    recall  f1-score   support

           0       0.49      0.76      0.60       253
           1       0.45      0.20      0.28       247

    accuracy                           0.48       500
   macro avg       0.47      0.48      0.44       500
weighted avg       0.47      0.48      0.44       500
```

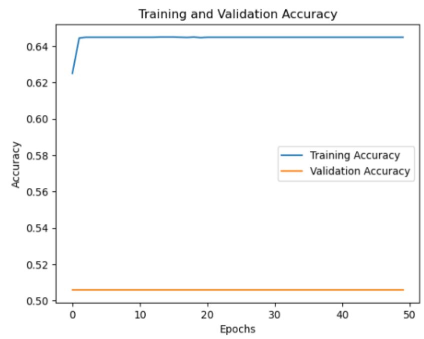
3. **Generate the following plots: [3 marks]**

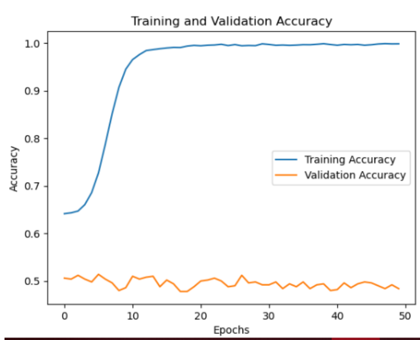
- **Loss plot - Training Loss and Validation Loss V/s Epochs.**
- **Accuracy plot - Training Accuracy, Validation Accuracy V/s Epochs**
- **Analyze and Explain the plots obtained**











The above graph for model 1 shows a lot of variation in validation loss which first increases then falls gradually whereas the training loss remains the same.The accuracy remains somewhat constant.

For graph 2, there are no such variations and it is almost constant.The accuracy is constant throughout.

For the third graph, there is variation in validation loss which increases then falls and the accuracy also shows a bit of variation.

4. **Report the overall Accuracy, Precision, Recall, F1 score for your test set. Also, report class-wise precision and recall and F1 score for test set. [3 marks]**

Model 1:

```
Following are the TEST metrics associated with the model

Test loss: 3.4917807579040527

Test acuuracy: 0.4950000047683716

Test precision: 0.4740484356880188

Test recall: 0.279591828584671

Test F1-Score: 0.34587135910987854

32/32 [==============================] - 0s 4ms/step
              precision    recall  f1-score   support

           0       0.50      0.70      0.59       510
           1       0.47      0.28      0.35       490

    accuracy                           0.49      1000
   macro avg       0.49      0.49      0.47      1000
weighted avg       0.49      0.49      0.47      1000
```

Model 2:

```
Test loss: 0.7309284210205078

Test acuuracy: 0.5099999904632568

Test precision: 0.0

Test recall: 0.0

Test F1-Score: 0.0

32/32 [==============================] - 0s 10ms/step
              precision    recall  f1-score   support

           0       0.51      1.00      0.68       510
           1       0.00      0.00      0.00       490

    accuracy                           0.51      1000
   macro avg       0.26      0.50      0.34      1000
weighted avg       0.26      0.51      0.34      1000
```

Model 3:

```
Following are the TEST metrics associated with the model

Test loss: 4.72576904296875

Test acuuracy: 0.5170000195503235

Test precision: 0.5140562057495117

Test recall: 0.2612244784832001

Test F1-Score: 0.3352556824684143

32/32 [==============================] - 0s 5ms/step
              precision    recall  f1-score   support

           0       0.52      0.76      0.62       510
           1       0.51      0.26      0.35       490

    accuracy                           0.52      1000
   macro avg       0.52      0.51      0.48      1000
weighted avg       0.52      0.52      0.48      1000
```

Model	Test Accuracy
1	49.5
2	50.9
3	**51.7**

Model 3 performed the best. Thought the accuracies are almost somewhat nearby, but still Model 3 is best.

TASK II (Unimodal: Text-Only) [20 marks]
Text-only hateful meme detection

Task: Develop text-only detection system to classify memes as hateful or not hateful.

1. **Data preparation: Preprocess the text extracted by cleaning, tokenizing, and converting it into a representation that can be used as input to the deep learning model. [2 marks]**
Pre- Process.
   1. Load the data.
   2. Define the max_words and max_len needed for texts.
   3. Tokenize the text.
   4. Convert it into sequences.

2. **Propose and implement a text-only model for classification using any deep learning model of your choice such as BERT, LSTM, XLNet, etc. [12 marks]**
   1. We have implemented a LSTM model. We have tried variations in the model to find the best one.

Model 1:

1. Input and Embedding Layer.
2. LSTM with Dense and Dropout of 0.2
3. Loss = binary_crossentropy, Optimizer = Adam. (lr = 0.001)

Model 2:

1. Input and Embedding Layer.
2. LSTM(128) + LSTM (64) with Dense and Dropout of 0.2
3. Loss = binary_crossentropy, Optimizer =RmsProp. (lr = 0.001)

Model 3:

1. Conv2D and Max Pooling
2. LSTM and Dropout of 0.2
3. Loss = binary_crossentropy, Optimizer =Adam. (lr = 0.001)

Model 4:

1. Pre-trained word embeddings (glove)
2. Input and Embedding Layer.
3. LSTM with Dense and Dropout of 0.2
4. Loss = binary_crossentropy, Optimizer = Adam. (lr = 0.001)

Different functions have been made for plotting graphs and calculating metrics.

Model 1 Results:

```
Following are the TEST metrics associated with the model

Test loss: 6.397130012512207

Test acuuracy: 0.5419999957084656

Test precision: 0.5533333420753479

Test recall: 0.33877551555633545

Test F1-Score: 0.41793137788772583

27/32 [========================>.....] - ETA: 0s
              precision    recall  f1-score   support

           0       0.96      0.89      0.92      5481
           1       0.82      0.93      0.87      3019

    accuracy                           0.90      8500
   macro avg       0.89      0.91      0.90      8500
weighted avg       0.91      0.90      0.91      8500
```

```
Following are the TRAIN metrics associated with the model

Train loss: 0.14656448364257812

Train acuuracy: 0.9044705629348755

Train precision: 0.8242726922035217

Train recall: 0.9291155934333801

Train F1-Score: 0.8632691502571106

266/266 [==============================] - 1s 4ms/step
              precision    recall  f1-score   support

           0       0.96      0.89      0.92      5481
           1       0.82      0.93      0.87      3019

    accuracy                           0.90      8500
   macro avg       0.89      0.91      0.90      8500
weighted avg       0.91      0.90      0.91      8500

######################################################################
########################
######################################################################
########################
Following are the DEV metrics associated with the model

Train loss: 6.821752548217773

Train acuuracy: 0.5220000147819519

Train precision: 0.5243902206420898

Train recall: 0.3481781482696533

Train F1-Score: 0.42092370986938477

16/16 [==============================] - 0s 4ms/step
              precision    recall  f1-score   support

           0       0.52      0.69      0.59       253
           1       0.52      0.35      0.42       247

    accuracy                           0.52       500
   macro avg       0.52      0.52      0.51       500
weighted avg       0.52      0.52      0.51       500
```

Model 2 Results:

```
Following are the TEST metrics associated with the model

Test loss: 2.3108389377593994

Test acuuracy: 0.49900001287460327

Test precision: 0.48927876353263855

Test recall: 0.5122448801994324

Test F1-Score: 0.49823644757270813

              precision    recall  f1-score   support

           0       0.51      0.49      0.50       510
           1       0.49      0.51      0.50       490

    accuracy                           0.50      1000
   macro avg       0.50      0.50      0.50      1000
weighted avg       0.50      0.50      0.50      1000
```

```
Following are the TRAIN metrics associated with the model

Train loss: 0.23462632298469543

Train acuuracy: 0.8947058916091919

Train precision: 0.7825971245765686

Train recall: 0.9741636514663696

Train F1-Score: 0.8579731583595276

266/266 [==============================] - 3s 11ms/step
              precision    recall  f1-score   support

           0       0.98      0.85      0.91      5481
           1       0.78      0.97      0.87      3019

    accuracy                           0.89      8500
   macro avg       0.88      0.91      0.89      8500
weighted avg       0.91      0.89      0.90      8500

################################################################
########################
################################################################
########################
Following are the DEV metrics associated with the model

Train loss: 2.705322742462158

Train acuuracy: 0.48399999737739563

Train precision: 0.4761904776096344

Train recall: 0.44534412026405334

Train F1-Score: 0.4609042704105377

16/16 [==============================] - 0s 11ms/step
              precision    recall  f1-score   support

           0       0.49      0.52      0.51       253
           1       0.48      0.45      0.46       247

    accuracy                           0.48       500
   macro avg       0.48      0.48      0.48       500
weighted avg       0.48      0.48      0.48       500
```

Model 3:

```
Following are the TEST metrics associated with the model

Test loss: 3.7579970359802246

Test acuuracy: 0.5360000133514404

Test precision: 0.5329949259757996

Test recall: 0.4285714328289032

Test F1-Score: 0.47045087814331055

              precision    recall  f1-score   support

           0       0.54      0.64      0.58       510
           1       0.53      0.43      0.48       490

    accuracy                           0.54      1000
   macro avg       0.54      0.53      0.53      1000
weighted avg       0.54      0.54      0.53      1000
```

```
Following are the TRAIN metrics associated with the model

Train loss: 0.14254863560199738

Train acuuracy: 0.9035294055938721

Train precision: 0.792497992515564

Train recall: 0.986750602722168

Train F1-Score: 0.8704138994216919

266/266 [==============================] - 1s 4ms/step
              precision    recall  f1-score   support

           0       0.99      0.86      0.92      5481
           1       0.79      0.99      0.88      3019

    accuracy                           0.90      8500
   macro avg       0.89      0.92      0.90      8500
weighted avg       0.92      0.90      0.91      8500

######################################################################
########################
######################################################################
########################
Following are the DEV metrics associated with the model

Train loss: 3.963120222091675

Train acuuracy: 0.5080000162124634

Train precision: 0.5027027130126953

Train recall: 0.37651821970939636

Train F1-Score: 0.43201911449432373

16/16 [==============================] - 0s 3ms/step
              precision    recall  f1-score   support

           0       0.51      0.64      0.57       253
           1       0.50      0.38      0.43       247

    accuracy                           0.51       500
   macro avg       0.51      0.51      0.50       500
weighted avg       0.51      0.51      0.50       500
```

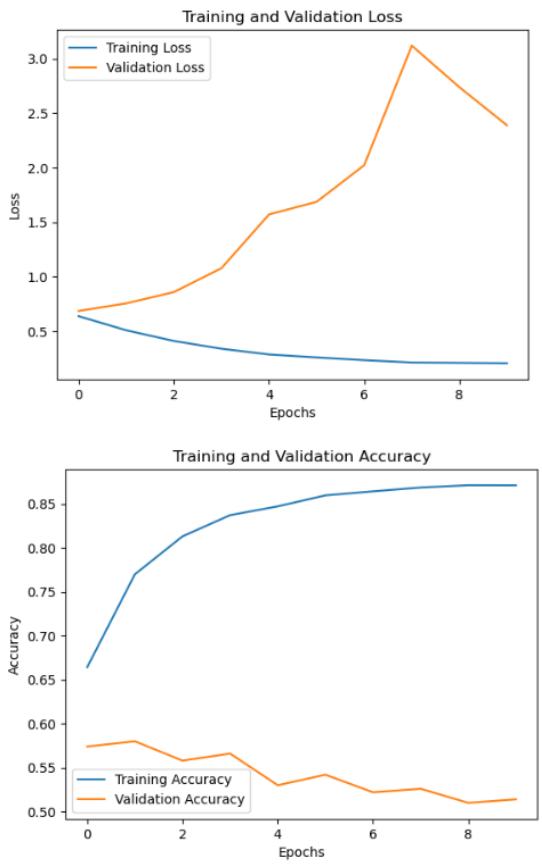
Model 4 Results:

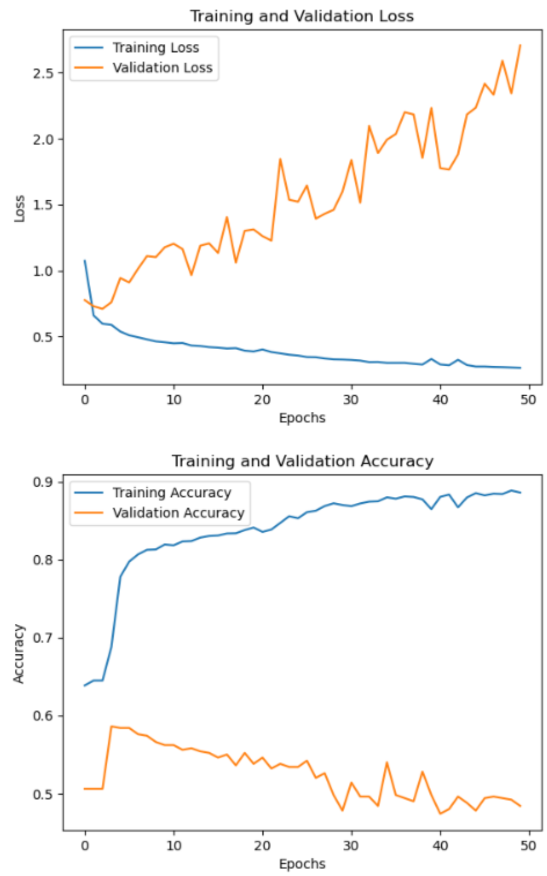
```
Following are the TEST metrics associated with the model

Test loss: 4.041996955871582

Test acuuracy: 0.5720000267028809

Test precision: 0.6156716346740723

Test recall: 0.33673468232154846

Test F1-Score: 0.42475008964538574
```

```
32/32 [==============================] - 1s 6ms/step
              precision    recall  f1-score   support

           0       0.56      0.80      0.66       510
           1       0.62      0.34      0.44       490

    accuracy                           0.57      1000
   macro avg       0.59      0.57      0.55      1000
weighted avg       0.59      0.57      0.55      1000
```

```
Following are the TRAIN metrics associated with the model

Train loss: 0.1687285602092743

Train acuuracy: 0.8962352871894836

Train precision: 0.7893311381340027

Train recall: 0.965551495552063

Train F1-Score: 0.8583505749702454

266/266 [==============================] - 2s 6ms/step
              precision    recall  f1-score   support

           0       0.98      0.86      0.91      5481
           1       0.79      0.97      0.87      3019

    accuracy                           0.90      8500
   macro avg       0.88      0.91      0.89      8500
weighted avg       0.91      0.90      0.90      8500

#####################################################################
########################
#####################################################################
########################
Following are the DEV metrics associated with the model

Train loss: 4.281875133514404

Train acuuracy: 0.5580000281333923

Train precision: 0.6048387289047241

Train recall: 0.30364373326301575

Train F1-Score: 0.39796292781829834

16/16 [==============================] - 0s 6ms/step
              precision    recall  f1-score   support

           0       0.54      0.81      0.65       253
           1       0.60      0.30      0.40       247

    accuracy                           0.56       500
   macro avg       0.57      0.55      0.53       500
weighted avg       0.57      0.56      0.53       500
```

3. **Generate the following plots: [3 marks]**
- **Loss plot - Training Loss and Validation Loss V/s Epochs.**
- **Accuracy plot - Training Accuracy, Validation Accuracy V/s Epochs**
- **Analyze and Explain the plots obtained**

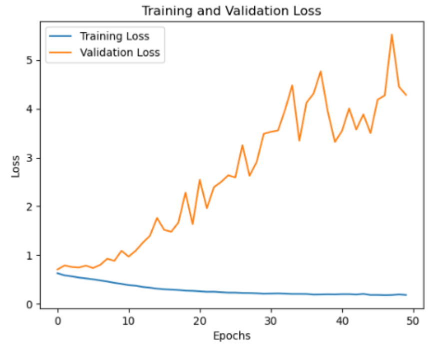


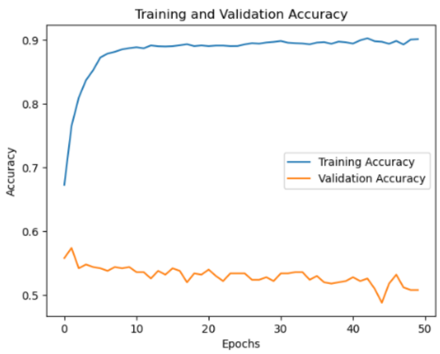


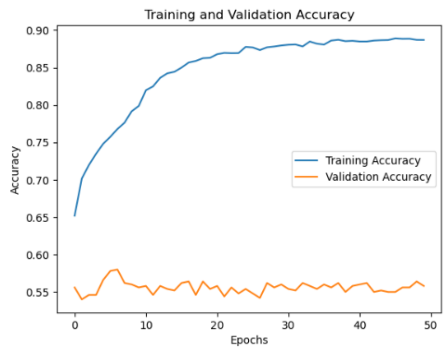
Moded 1 Model 2

Model 1 plot shows that the training loss gradually decreases whereas the validation loss first increased then decreased.
The training accuracy achieved is good whereas the validation accuracy falls gradually.

Model 2 plot shows a lot of variation in the loss and accuracy curves. The loss curve validation loss increases but the training loss falls gradually.
The validation accuracy first increased but then fell.







Model 3 plot shows that the train loss almost reamined the same whereas the validation oss shows variations and increased but then fell. The accuracy remains somewhat same.
Model 4 graphs shows that the model performed best as the accuracy is better than the others. Loss increases gradually. Validation accuracy remains somewhat constant.

4. **Report the over Accuracy, Precision, Recall, and F1 score for your test set. Also, report class-wise precision and recall and F1 score for the test set. [3 marks]**

Model 1:

```
Following are the TEST metrics associated with the model

Test loss: 6.397130012512207

Test acuuracy: 0.5419999957084656

Test precision: 0.5533333420753479

Test recall: 0.33877551555633545

Test F1-Score: 0.41793137788772583

27/32 [========================>.....] - ETA: 0s
              precision    recall  f1-score   support

           0       0.96      0.89      0.92      5481
           1       0.82      0.93      0.87      3019

    accuracy                           0.90      8500
   macro avg       0.89      0.91      0.90      8500
weighted avg       0.91      0.90      0.91      8500
```

Model 2:

```
Following are the TEST metrics associated with the model

Test loss: 2.3108389377593994

Test acuuracy: 0.49900001287460327

Test precision: 0.48927876353263855

Test recall: 0.5122448801994324

Test F1-Score: 0.49823644757270813
              precision    recall  f1-score   support

           0       0.51      0.49      0.50       510
           1       0.49      0.51      0.50       490

    accuracy                           0.50      1000
   macro avg       0.50      0.50      0.50      1000
weighted avg       0.50      0.50      0.50      1000
```

Model 3:

```
Following are the TEST metrics associated with the model

Test loss: 3.7579970359802246

Test acuuracy: 0.5360000133514404

Test precision: 0.5329949259757996

Test recall: 0.4285714328289032

Test F1-Score: 0.47045087814331055
              precision    recall  f1-score   support

           0       0.54      0.64      0.58       510
           1       0.53      0.43      0.48       490

    accuracy                           0.54      1000
   macro avg       0.54      0.53      0.53      1000
weighted avg       0.54      0.54      0.53      1000
```

Model 4:

```
Following are the TEST metrics associated with the model

Test loss: 4.041996955871582

Test acuuracy: 0.5720000267028809

Test precision: 0.6156716346740723

Test recall: 0.33673468232154846

Test F1-Score: 0.42475008964538574
```

```
32/32 [==============================] - 1s 6ms/step
              precision    recall  f1-score   support

           0       0.56      0.80      0.66       510
           1       0.62      0.34      0.44       490

    accuracy                           0.57      1000
   macro avg       0.59      0.57      0.55      1000
weighted avg       0.59      0.57      0.55      1000
```

Summary 👍The best model comes out to be Model 4 in which we used pre -trained word embeddings.

Model	**Test Accuracy**
1	54.1
2	49.9
3	53.6
4	**57.2**

TASK III (Multimodal: Image+ Text-Based Classification) [30 marks]

Multimodal (Visuals & Language) hateful meme detection:

1. **Select an appropriate deep learning model architecture for joint image and text classification, such as a multimodal fusion model (early or late fusion). Apply appropriate preprocessing. You can employ any combination of CNN, LSTM, or pretrained transformer models or use some multimodal model directly. Your base image, text model, and fusion technique should be clear.In report, explain in 3-4 lines along with a figure, the proposed architecture to handle multi-modal data. Your multimodal system should perform better in terms of Accuracy and F1 score than both image-only and text-only models.**
[10+4+2 marks] for the proposed multimodal model, improvement over unimodal and explanation respectively.
   1. The pre-processing steps are the same as above.
   2. We have use CNN_LSTM and VGG_LSTM model. CNN, VGG for the image task and LSTM for the text part.
   3. We have compared the variations in the models and have selected the best.

Epochs = 50
Model 1:
CNN
Conv2D with MaxPooling layers. (input is 224,224,3)
LSTM
INput and Embedding layer with Dropout = 0.5.
Loss = binary_crossentropy
Optimizer = Adam (lr=0.001)
Combine the image and text features.

Model 2:
VGG
LSTM
INput and Embedding matrix created with Dropout = 0.5.
Glove embedding.
Loss = binary_crossentropy
Optimizer = Adam (lr=0.001)
Combine the image and text features.

Model 3: (Best Model)
VGG
LSTM
INput and Embedding layer used with Dropout = 0.5.
Glove embedding.
Loss = binary_crossentropy Optimizer = Adam (lr=0.001)

Combine the image and text features.

```
Model: "model_1"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_5 (InputLayer)           [(None, 224, 224, 3  0           []
                                )]

 input_6 (InputLayer)           [(None, 256)]        0           []

 vgg16 (Functional)             (None, 7, 7, 512)    14714688    ['input_5[0][0]']

 embedding_1 (Embedding)        (None, 256, 128)     1536000     ['input_6[0][0]']

 flatten_1 (Flatten)            (None, 25088)        0           ['vgg16[0][0]']

 lstm_1 (LSTM)                  (None, 32)           20608       ['embedding_1[0][0]']

 concatenate_1 (Concatenate)    (None, 25120)        0           ['flatten_1[0][0]',
                                                                  'lstm_1[0][0]']

 dense_2 (Dense)                (None, 64)           1607744     ['concatenate_1[0][0]']

 dropout_1 (Dropout)            (None, 64)           0           ['dense_2[0][0]']

 dense_3 (Dense)                (None, 1)            65          ['dropout_1[0][0]']

==================================================================================================
Total params: 17,879,105
Trainable params: 3,164,417
Non-trainable params: 14,714,688
```

Different functions have been made for plotting graphs and metrics calculations.

Model 1:

```
Train loss: 0.009517771191895008

Train acuuracy: 0.9977647066116333

Train precision: 0.998670220375061

Train recall: 0.995031476020813

Train F1-Score: 0.997039794921875

266/266 [==============================] - 2s 7ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5481
           1       1.00      1.00      1.00      3019

    accuracy                           1.00      8500
   macro avg       1.00      1.00      1.00      8500
weighted avg       1.00      1.00      1.00      8500
```

This is the Dev loss (by mistake printing statement says Train)

```
Train loss: 7.226615905761719

Train acuuracy: 0.527999997138977

Train precision: 0.5407407283782959

Train recall: 0.2955465614795685

Train F1-Score: 0.3777009844779968

16/16 [==============================] - 0s 7ms/step
              precision    recall  f1-score   support

           0       0.52      0.75      0.62       253
           1       0.54      0.30      0.38       247

    accuracy                           0.53       500
   macro avg       0.53      0.53      0.50       500
weighted avg       0.53      0.53      0.50       500
```

Model 2

```
Train loss: 0.6504706740379333

Train acuuracy: 0.6449411511421204

Train precision: 1.0

Train recall: 0.00033123549656011164

Train F1-Score: 0.0006265663541853428

              precision    recall  f1-score   support

           0       0.64      1.00      0.78      5481
           1       1.00      0.00      0.00      3019

    accuracy                           0.64      8500
   macro avg       0.82      0.50      0.39      8500
weighted avg       0.77      0.64      0.51      8500
```

```
Train loss: 0.7333693504333496

Train acuuracy: 0.5059999823570251

Train precision: 0.0

Train recall: 0.0

Train F1-Score: 0.0

16/16 [==============================] - 1s 36ms/step
              precision    recall  f1-score   support

           0       0.51      1.00      0.67       253
           1       0.00      0.00      0.00       247

    accuracy                           0.51       500
   macro avg       0.25      0.50      0.34       500
weighted avg       0.26      0.51      0.34       500
```

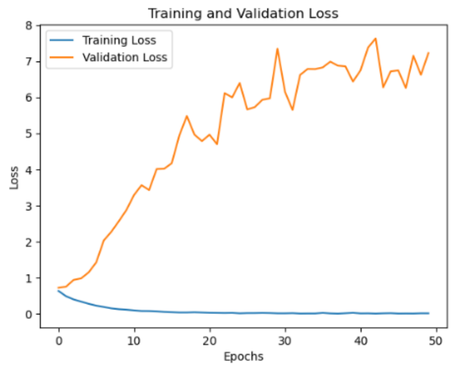
Model 3:

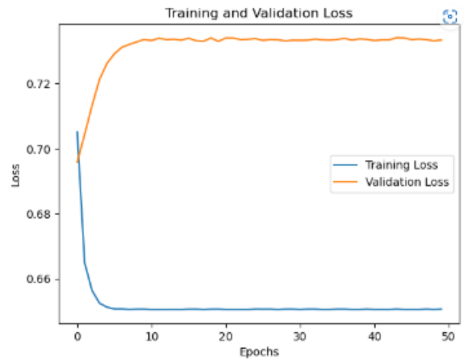
```
Train loss: 0.048057109117507935

Train acuuracy: 0.9784705638885498

Train precision: 0.9467548727989197

Train recall: 0.9953626990318298

Train F1-Score: 0.9680393934249878

266/266 [==============================] - 10s 36ms/step
              precision    recall  f1-score   support

           0       1.00      0.97      0.98      5481
           1       0.95      1.00      0.97      3019

    accuracy                           0.98      8500
   macro avg       0.97      0.98      0.98      8500
weighted avg       0.98      0.98      0.98      8500
```

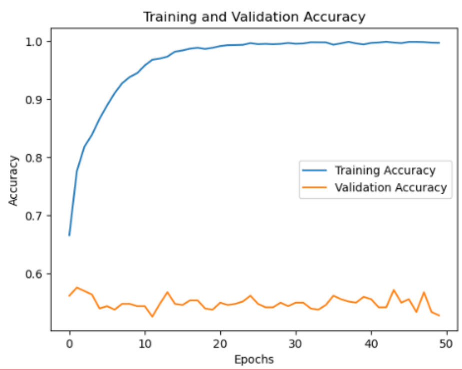
```
Train loss: 3.758172035217285

Train acuuracy: 0.5680000185966492

Train precision: 0.6115108132362366

Train recall: 0.3441295623779297

Train F1-Score: 0.4410113990306854

16/16 [==============================] - 1s 36ms/step
              precision    recall  f1-score   support

           0       0.55      0.79      0.65       253
           1       0.61      0.34      0.44       247

    accuracy                           0.57       500
   macro avg       0.58      0.57      0.54       500
weighted avg       0.58      0.57      0.55       500
```

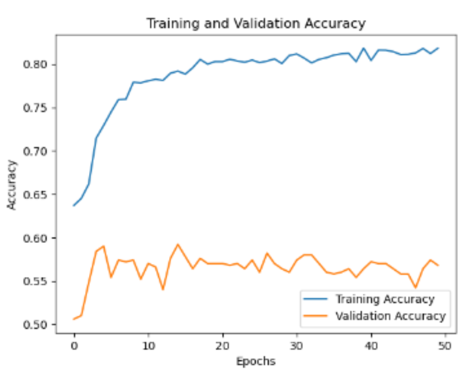
2. **Generate the following plots: [3 marks]**

- **Loss plot - Training Loss and Validation Loss V/s Epochs.**
- **Accuracy plot - Training Accuracy, Validation Accuracy V/s Epochs**
- **Analyze and Explain the plots obtained**









The above plot for Model 1 shows variations in the validation loss which gradually increases. The accuracy is somewhat constant throughout and is less.

There are very less variations for the plots of model 2 and the loss and accuarcies are almost constant throughout.

Model 3 shows the best accuracy Training as well as Validation. Though there are variations but the model outperforms the other models.

3. **Report the overall Accuracy, Precision, Recall, F1 score for your test set. Also, report class-wise precision and recall and F1 score for test set. [3 marks]**

Model 1:

```
Following are the TEST metrics associated with the model

Test loss: 7.324632167816162

Test acuuracy: 0.5559999942779541

Test precision: 0.5958333611488342

Test recall: 0.2918367385864258

Test F1-Score: 0.38176876306533813

              precision    recall  f1-score   support

           0       0.54      0.81      0.65       510
           1       0.60      0.29      0.39       490

    accuracy                           0.56      1000
   macro avg       0.57      0.55      0.52      1000
weighted avg       0.57      0.56      0.52      1000
```

Model 2:

```
Following are the TEST metrics associated with the model

Test loss: 0.7309843897819519

Test acuuracy: 0.5099999904632568

Test precision: 0.0

Test recall: 0.0

Test F1-Score: 0.0


              precision    recall  f1-score   support

           0       0.51      1.00      0.68       510
           1       0.00      0.00      0.00       490

    accuracy                           0.51      1000
   macro avg       0.26      0.50      0.34      1000
weighted avg       0.26      0.51      0.34      1000
```

Model 3:

```
Following are the TEST metrics associated with the model

Test loss: 3.3704614639282227

Test acuuracy: 0.5770000219345093

Test precision: 0.6254681944847107

Test recall: 0.34081631898880005

Test F1-Score: 0.4235880970954895
```

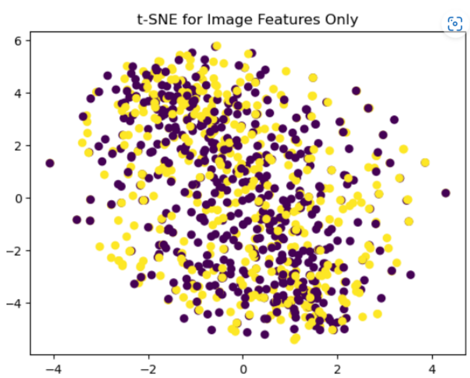
```
              precision    recall  f1-score   support

           0       0.56      0.80      0.66       510
           1       0.63      0.34      0.44       490

    accuracy                           0.58      1000
   macro avg       0.59      0.57      0.55      1000
weighted avg       0.59      0.58      0.55      1000
```

Model	Test Accuracy
1	55.5
2	50.9
3	**57.7**

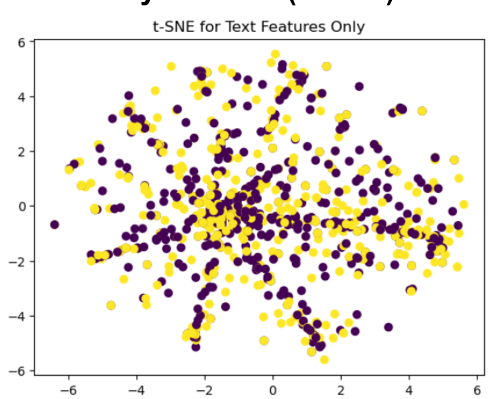
Models 1 and 3 are somewhat close but Model 3 performs the best.(VGG_LSTM)

4. **Visualize the following high-dimensional features into lower-dimensional space for hateful/not hateful classification in test set using T-SNE plots. You can subsample few test samples for this part and this part only, use atleast 50 hate, 50 non-hate samples but same samples used in all 3 model comparison. The features obtained here will be the last embedding layer of your respective task model (before classification layer):**

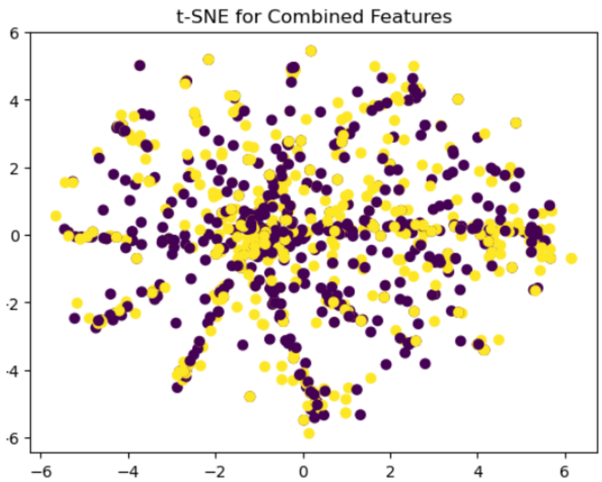
- **Image-only features (Task I)**



- **Text-only features (Task II)**



● **Joint Multi-modal (Text + Image) features (Task III)**



What can be said about the results obtained from unimodal models (Task I & II) vs multi-modal
model (Task III)? [8 marks] 2 for each plot + 2 for overall explanation.

Explanation:

This multimodal model performs better than both image-only and text-only models because it can leverage the information from both modalities to make a better decision. This model's accuracy and F1 score is higher than the unimodal models.

Equal contributions have been done in coding and making of report!