Agenda

- Goal
- Background and motivation
- Model & Network Structure
- Data preprocessing
- Network Training
- Result
- Future Work

Goal

- Apply transfer learning to predict the facial emotion
- 3 classes(Neutral, Happy, sad)
- Training data : 15k image
- Validation data: 5k image

Background and Motivation

Emotions are a person’s emotional and psychological changes with different situation.

Detecting Emotions would be huge step Artificial Intelligence

Model & Network Structure

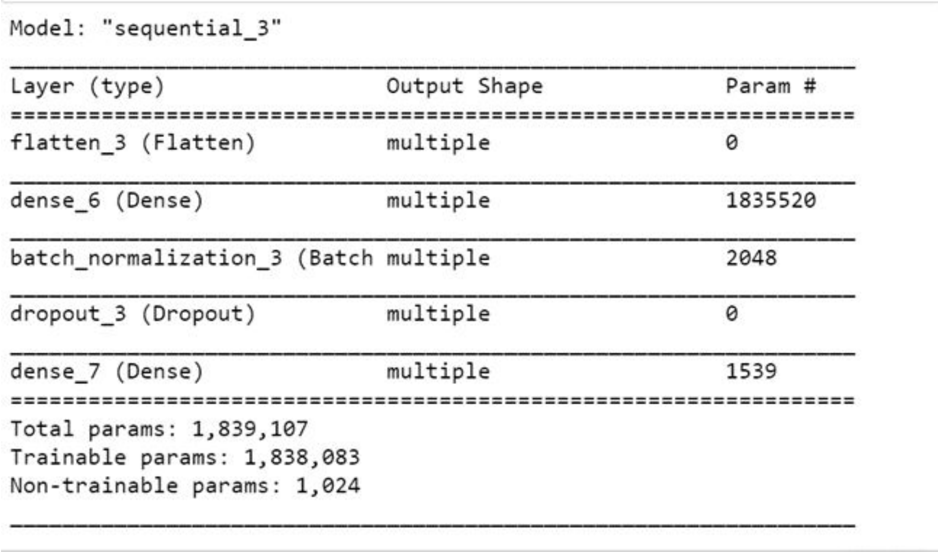
- Consist of VGG16, DenseNet121 and CNN

- Use VGG16 and Densnet121 as feature extractor

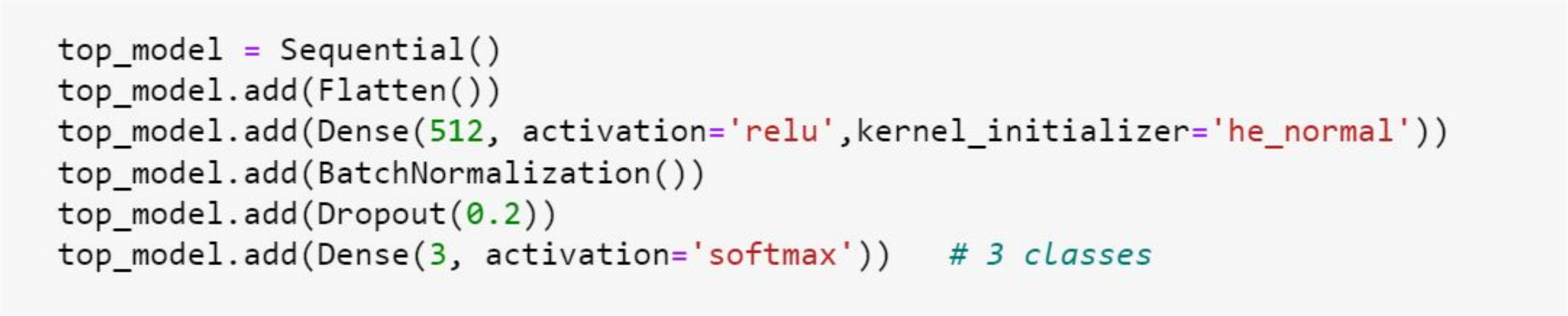
- Both VGG16 and Densent121 connect to same CNN

Model & Network Structure(contd.)

CNN



```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
flatten_3 (Flatten)          multiple                  0         
_________________________________________________________________
dense_6 (Dense)              multiple                  1835520   
_________________________________________________________________
batch_normalization_3 (Batch multiple                  2048      
_________________________________________________________________
dropout_3 (Dropout)          multiple                  0         
_________________________________________________________________
dense_7 (Dense)              multiple                  1539      
=================================================================
Total params: 1,839,107
Trainable params: 1,838,083
Non-trainable params: 1,024
_________________________________________________________________
```

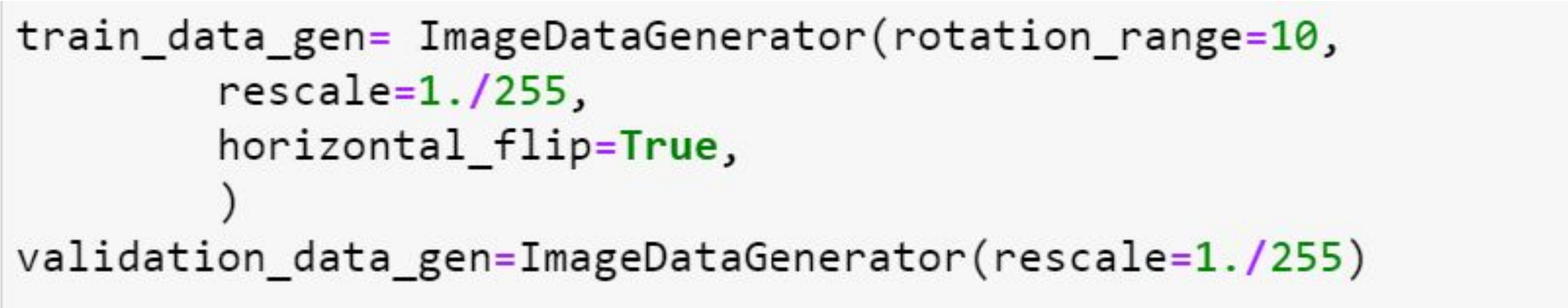


```
top_model = Sequential()
top_model.add(Flatten())
top_model.add(Dense(512, activation='relu',kernel_initializer='he_normal'))
top_model.add(BatchNormalization())
top_model.add(Dropout(0.2))
top_model.add(Dense(3, activation='softmax'))    # 3 classes
```

Data Preprocessing

- horizontal flip

- rescale



```
train_data_gen= ImageDataGenerator(rotation_range=10,
          rescale=1./255,
          horizontal_flip=True,
          )
validation_data_gen=ImageDataGenerator(rescale=1./255)
```

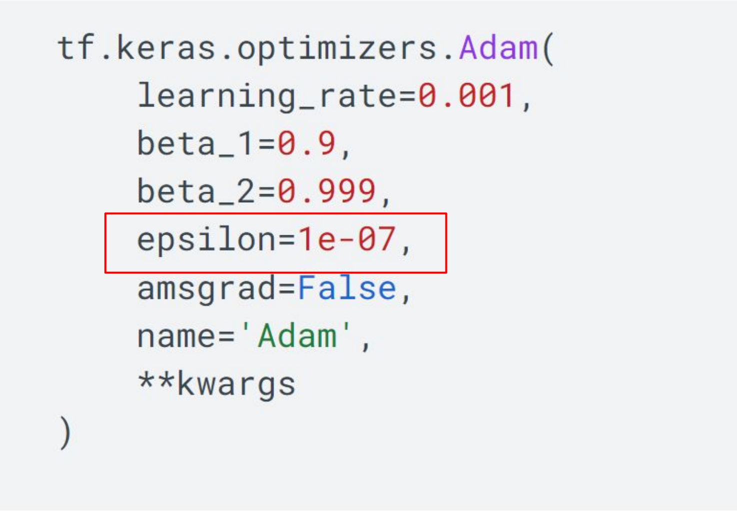
Network Training

Hyper parameter Tuning

1. Weight initialization method (with or without “he_normal”)(He et al., 2015).

2. Learning rate

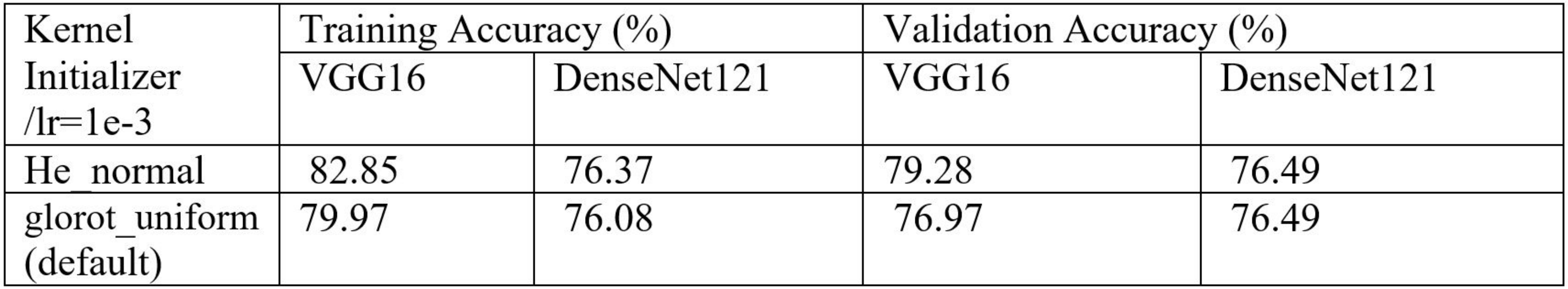
3. “Epsilon” in Adam (Reddi et.al, 2018)



```
tf.keras.optimizers.Adam(
    learning_rate=0.001,
    beta_1=0.9,
    beta_2=0.999,
    epsilon=1e-07,
    amsgrad=False,
    name='Adam',
    **kwargs
)
```

Result

Weight Initialization



Kernel Initializer /lr=1e-3	Training Accuracy (%)		Validation Accuracy (%)	
	VGG16	DenseNet121	VGG16	DenseNet121
He_normal	82.85	76.37	79.28	76.49
glorot_uniform (default)	79.97	76.08	76.97	76.49

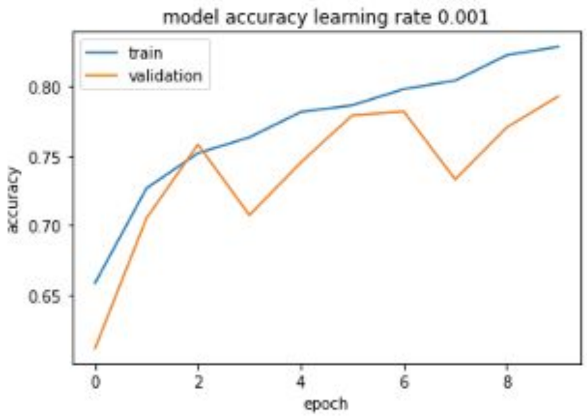
able1. Training accuracy and validation accuracy on weight initialization.

Result

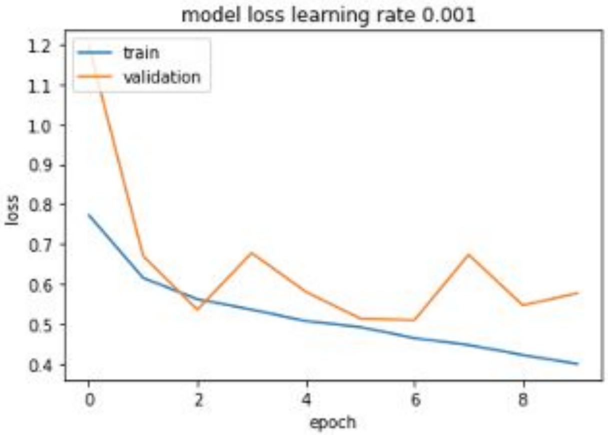
Weight Initialization

Network: VGG16 and CNN

Using “he_normal”

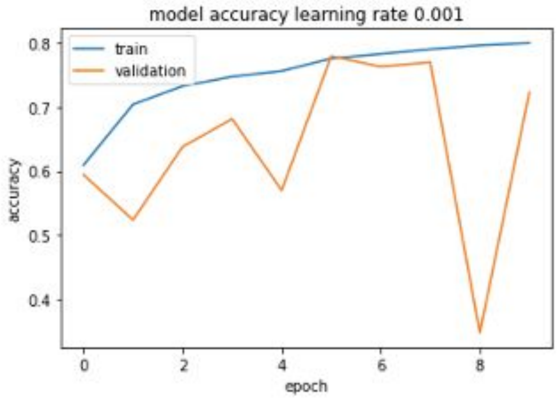


model accuracy learning rate 0.001

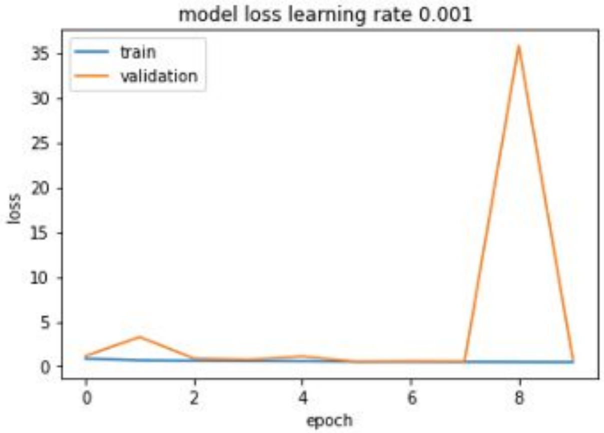


model loss learning rate 0.001

using “glorot_uniform”



model accuracy learning rate 0.001

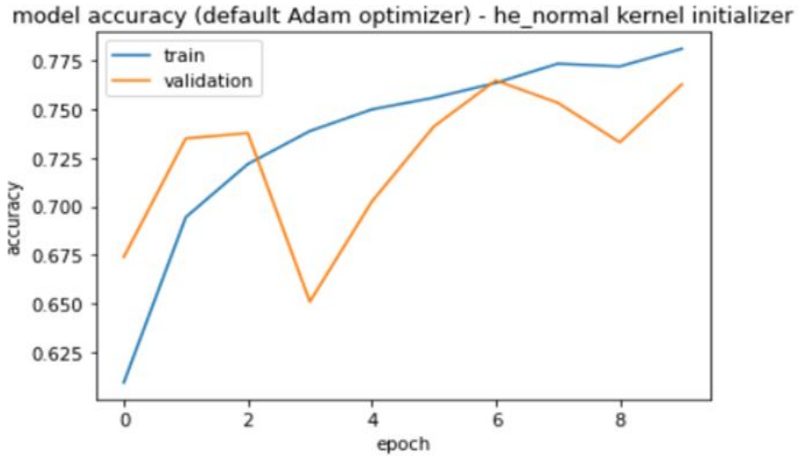


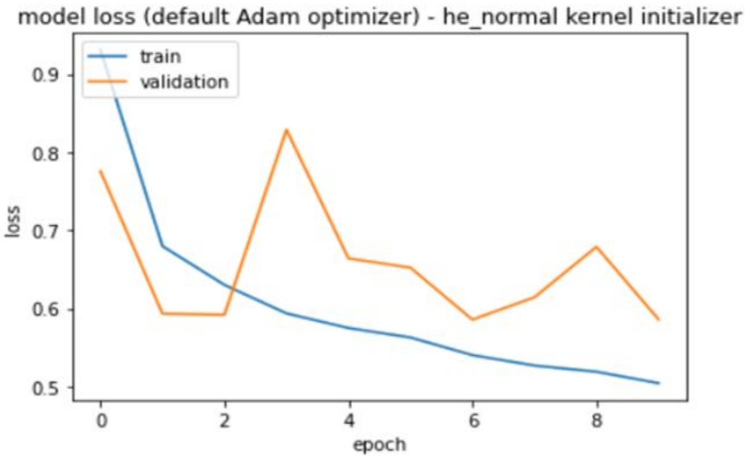
model loss learning rate 0.001

Result

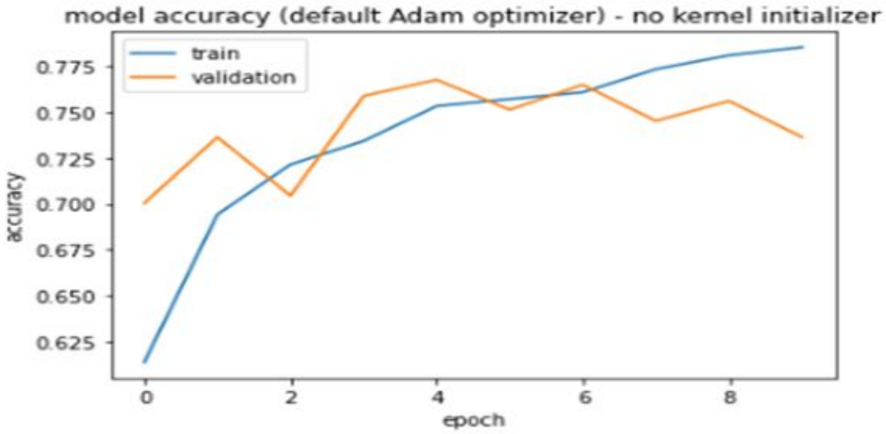
Weight Initialization Network: DenseNet 1212and CNN

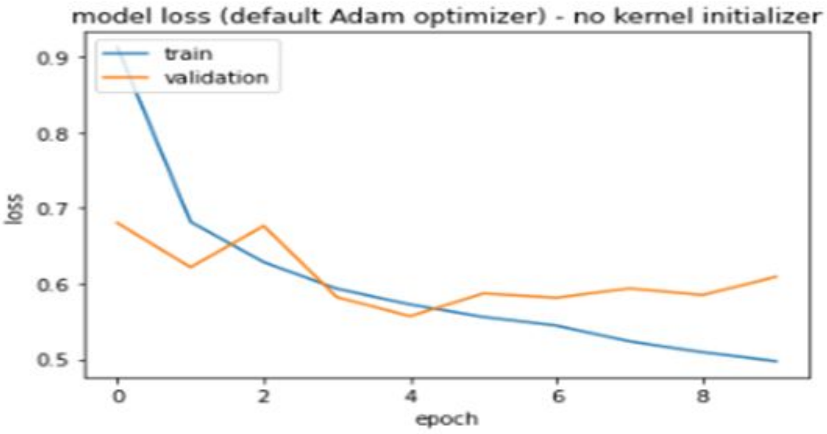
Using “he_normal”


model accuracy (default Adam optimizer) - he_normal kernel initializer


model loss (default Adam optimizer) - he_normal kernel initializer

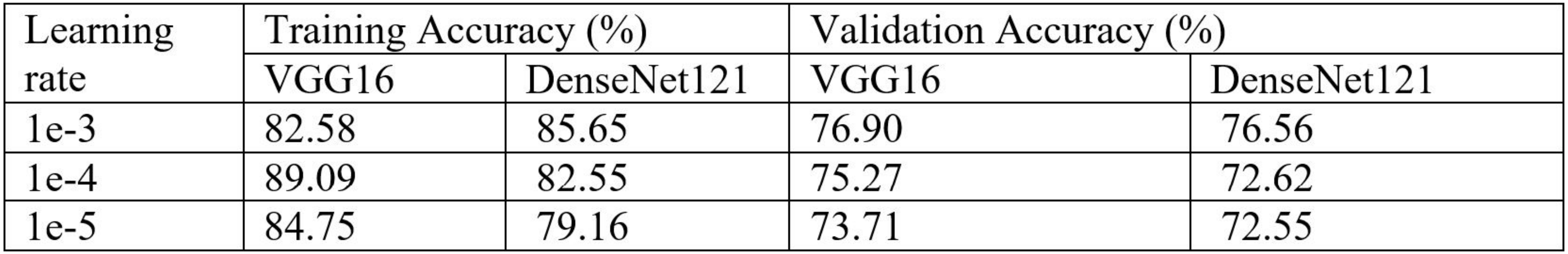
using “glorot_uniform”


model accuracy (default Adam optimizer) - no kernel initializer


model loss (default Adam optimizer) - no kernel initializer

Result(contd.)

Learning rate (using “he_normal”)



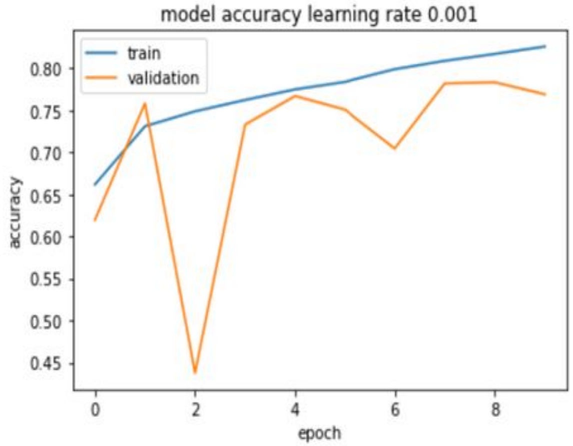
Learning rate	Training Accuracy (%)		Validation Accuracy (%)	
	VGG16	DenseNet121	VGG16	DenseNet121
1e-3	82.58	85.65	76.90	76.56
1e-4	89.09	82.55	75.27	72.62
1e-5	84.75	79.16	73.71	72.55

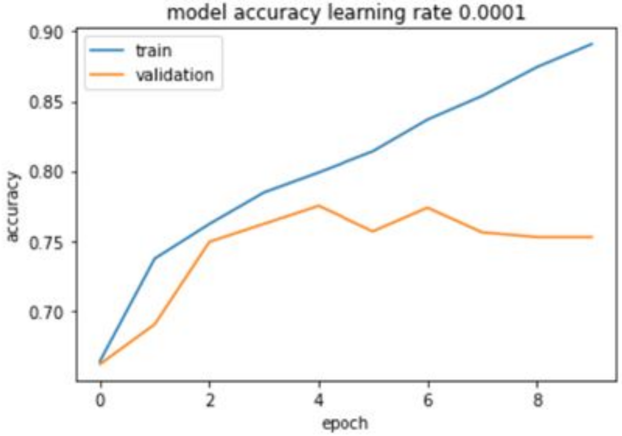
Table2. Training accuracy and validation accuracy on different learning rate

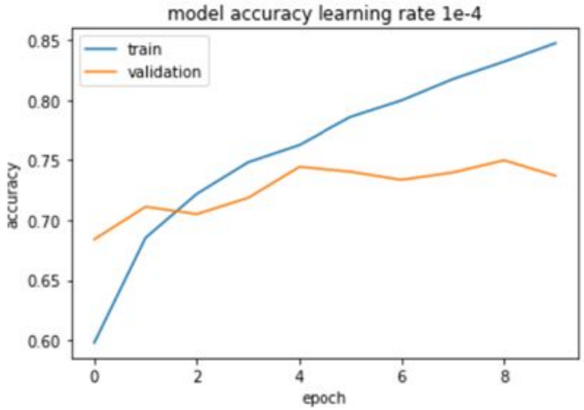
Result(contd.)

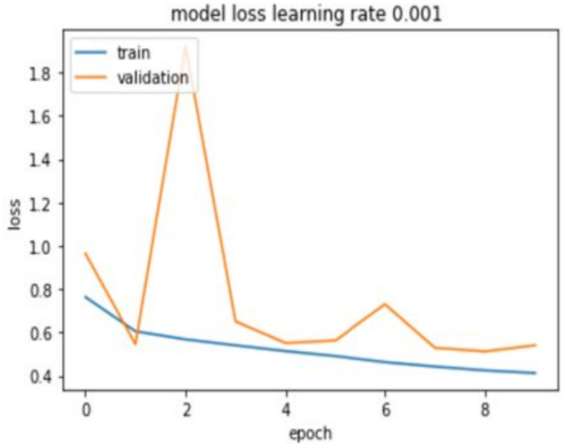
Learning rate

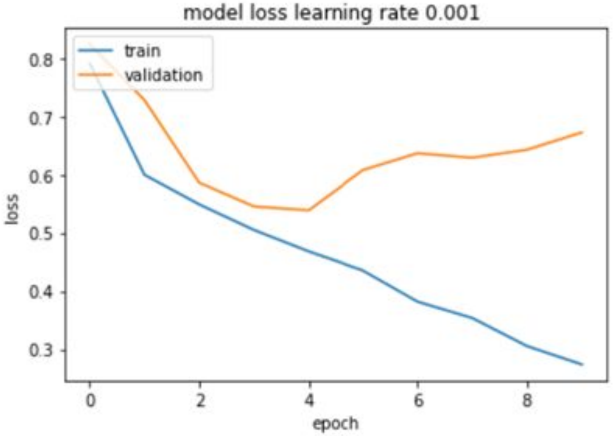
Network: VGG16 and CNN

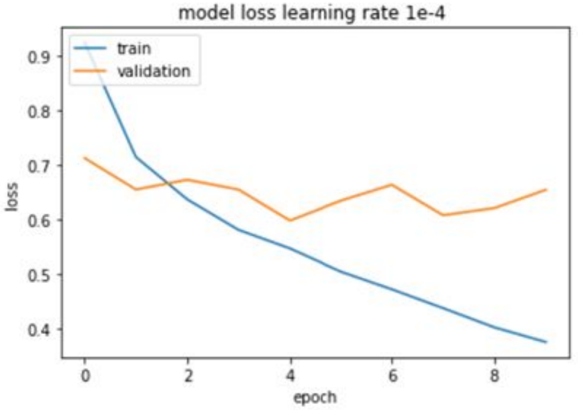

model accuracy learning rate 0.001


model accuracy learning rate 0.0001


model accuracy learning rate 1e-4


model loss learning rate 0.001

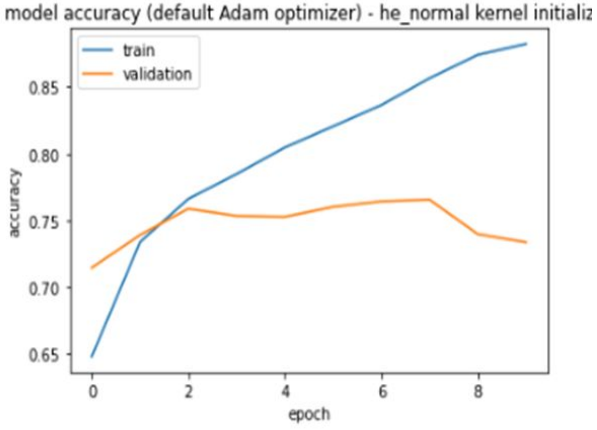

model loss learning rate 0.001

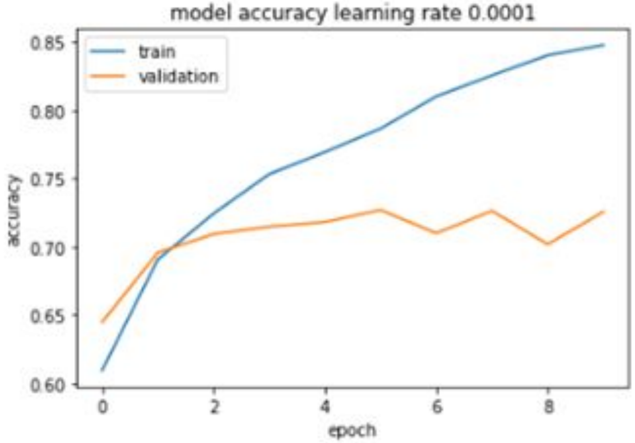

model loss learning rate 1e-4

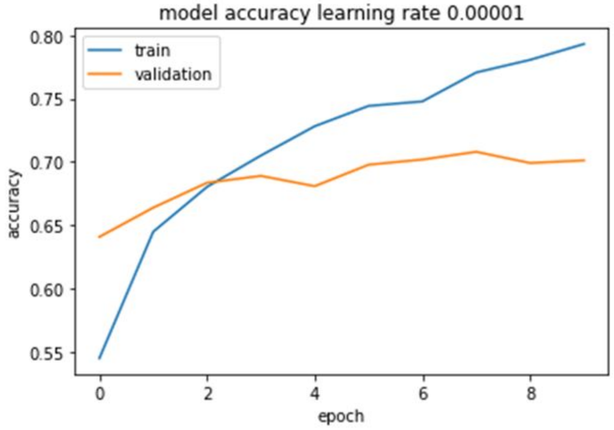
Result(contd.)

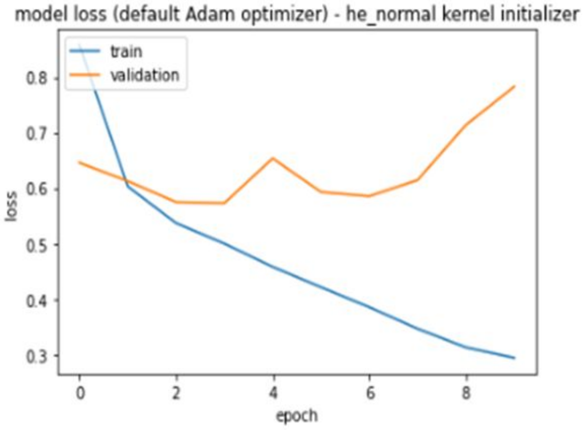
Learning rate

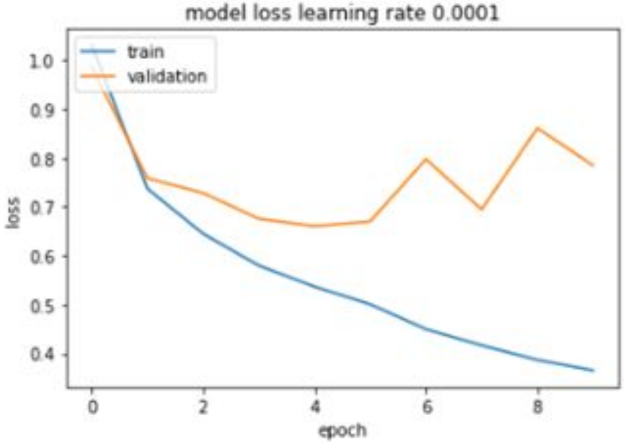
Network: DenseNet121 and CNN

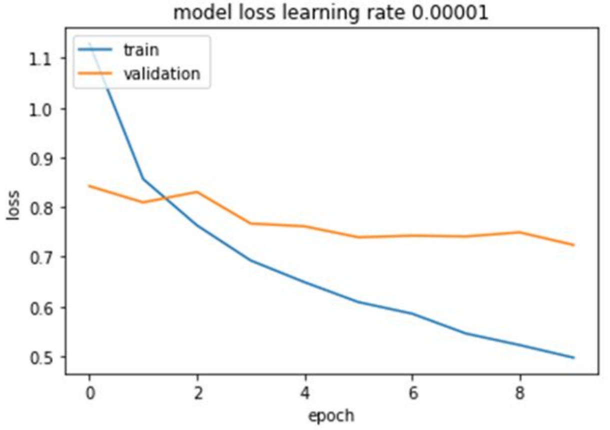






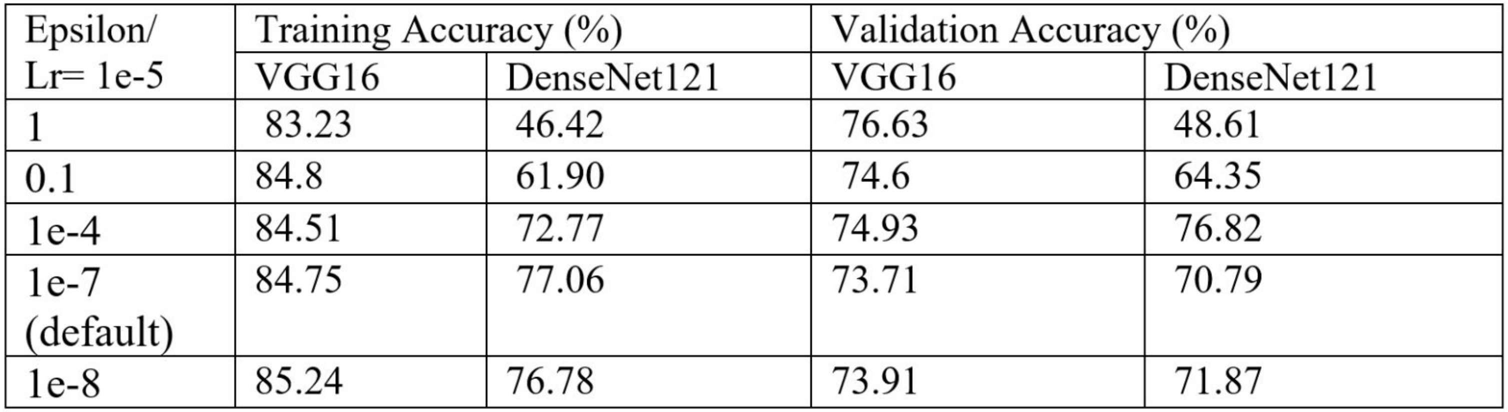






Result(contd.)

Epsilon



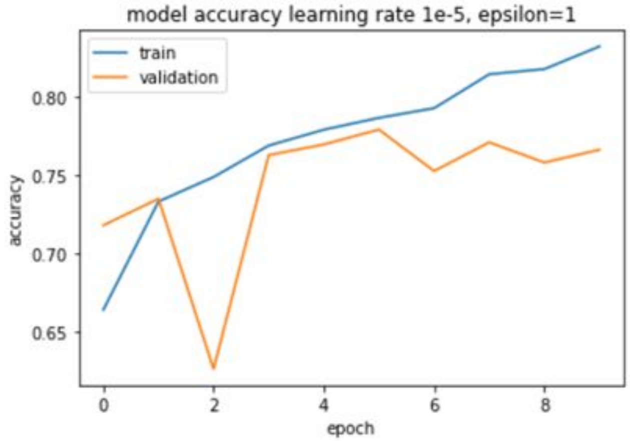
Epsilon/ Lr= 1e-5	Training Accuracy (%)		Validation Accuracy (%)	
	VGG16	DenseNet121	VGG16	DenseNet121
1	83.23	46.42	76.63	48.61
0.1	84.8	61.90	74.6	64.35
1e-4	84.51	72.77	74.93	76.82
1e-7 (default)	84.75	77.06	73.71	70.79
1e-8	85.24	76.78	73.91	71.87

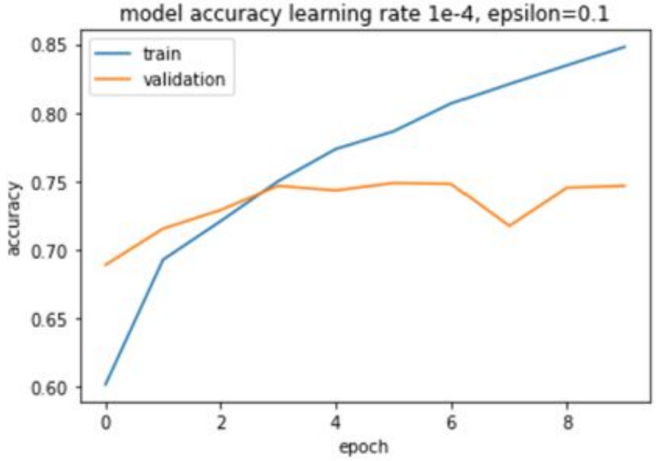
Table2. Training accuracy and validation accuracy on different epsilon value

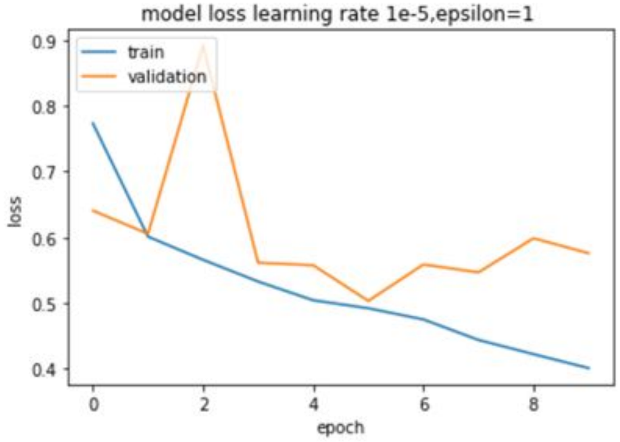
Result(contd.)

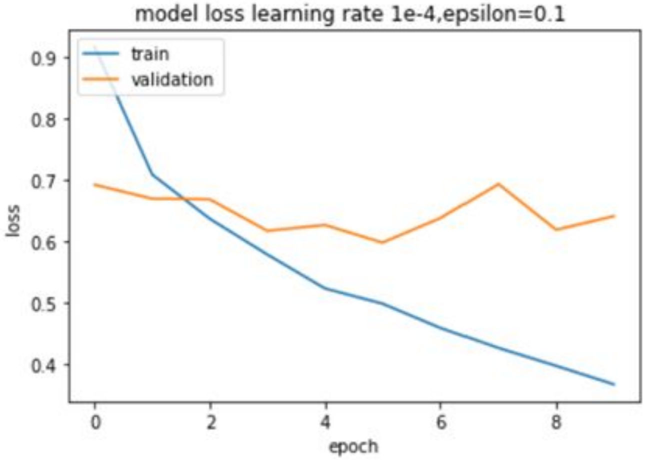
Epsilon

Network: VGG16 and CNN


model accuracy learning rate 1e-5, epsilon=1


model accuracy learning rate 1e-4, epsilon=0.1

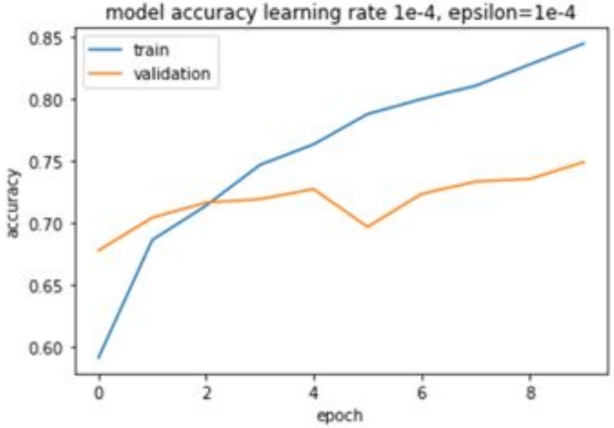

model loss learning rate 1e-5,epsilon=1

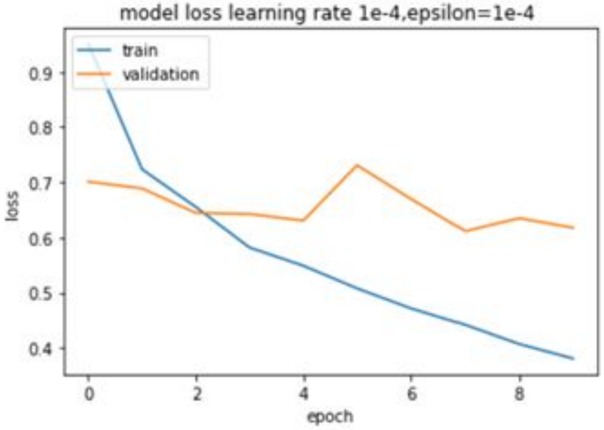

model loss learning rate 1e-4,epsilon=0.1

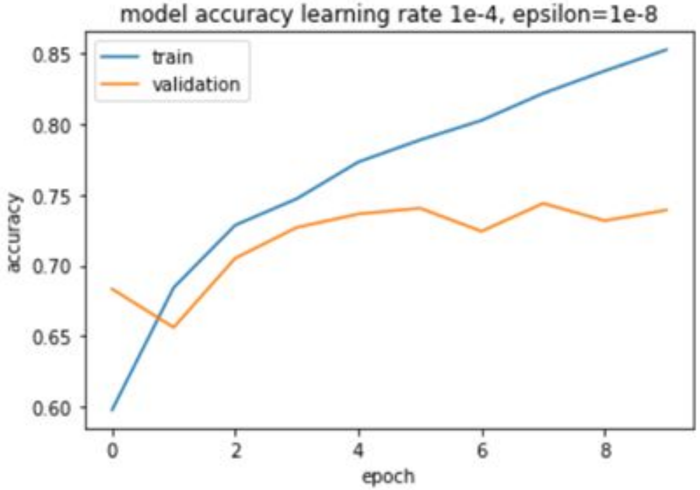
Result(contd.)

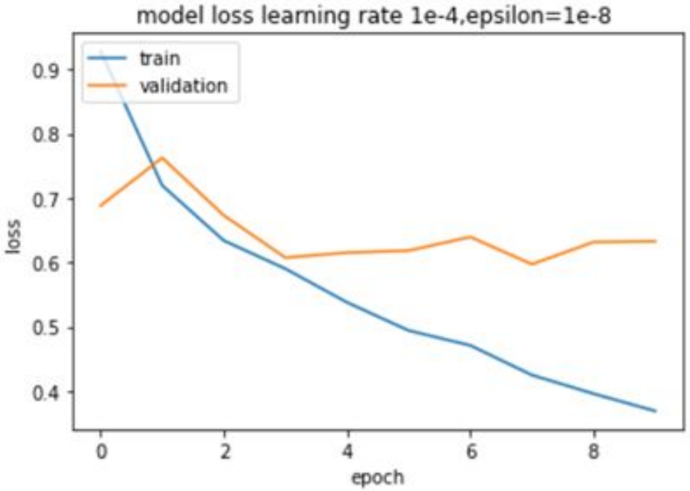
Epsilon

Network: VGG16 and CNN


model accuracy learning rate 1e-4, epsilon=1e-4


model loss learning rate 1e-4,epsilon=1e-4

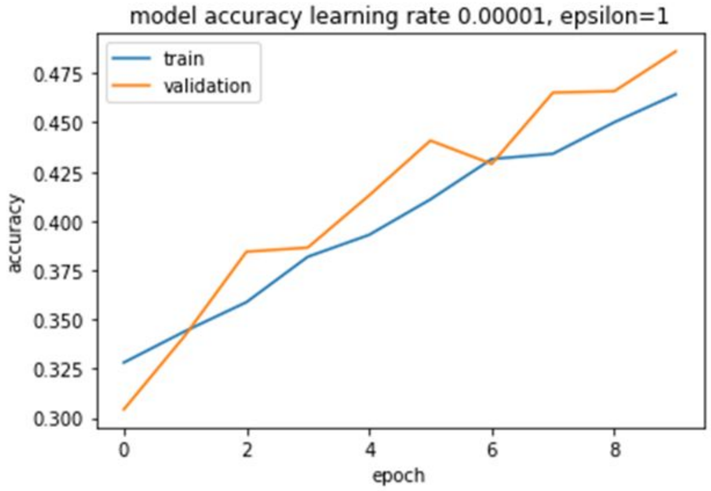

model accuracy learning rate 1e-4, epsilon=1e-8

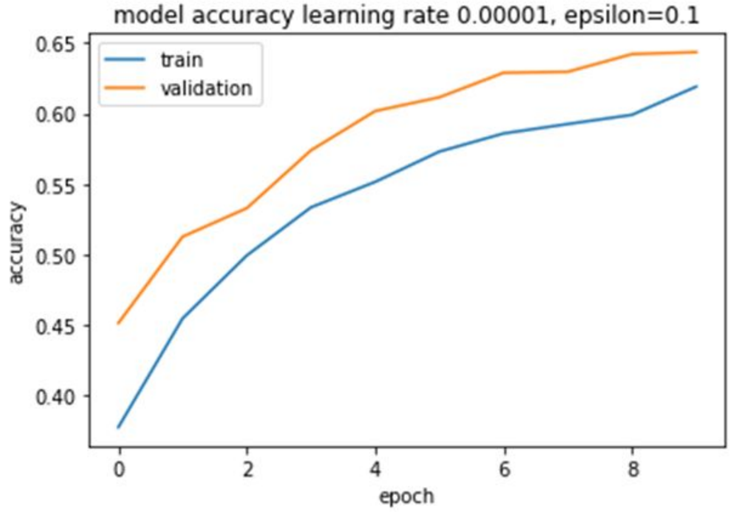

model loss learning rate 1e-4,epsilon=1e-8

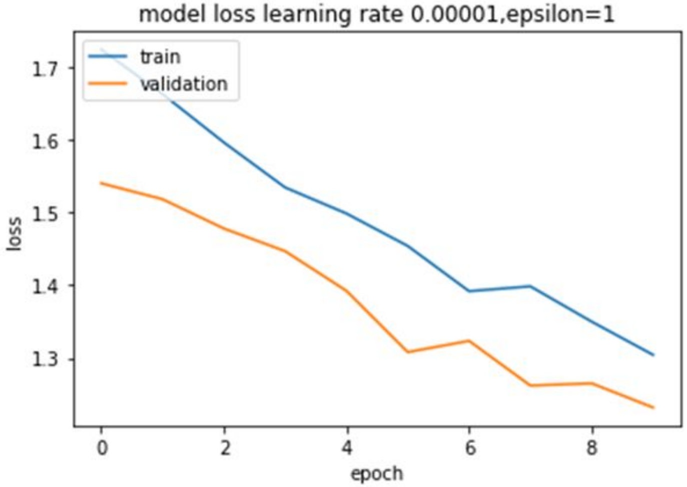
Result(contd.)

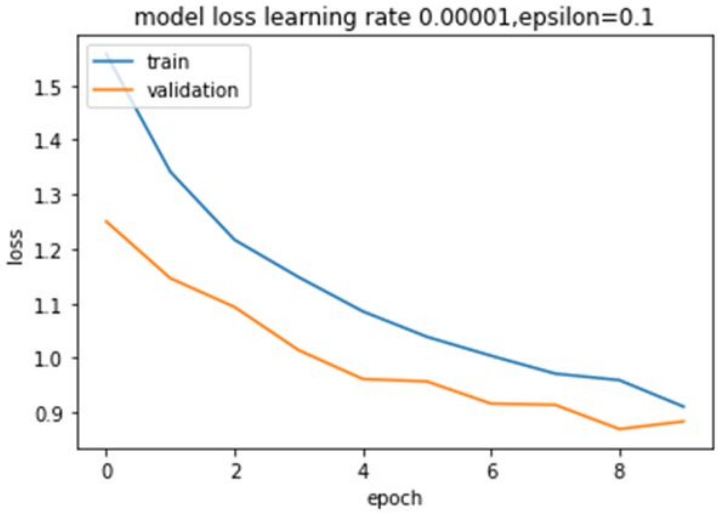
Epsilon

Network: DenseNet121 and CNN





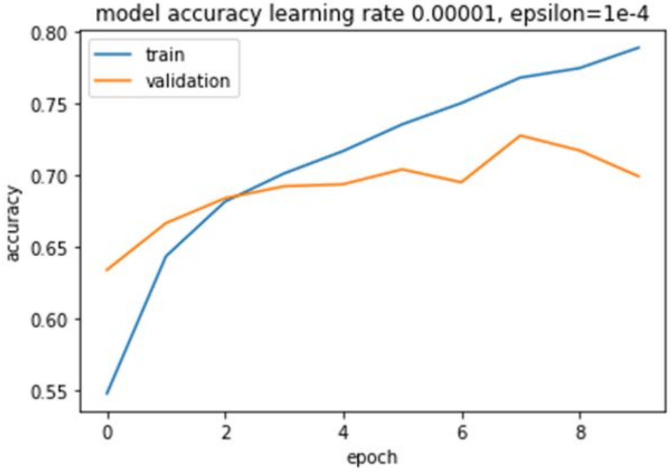


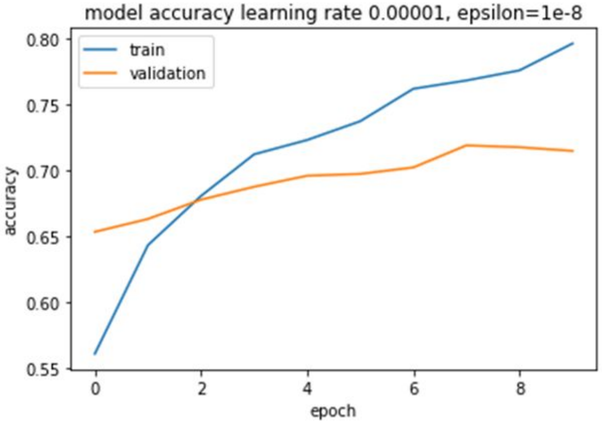


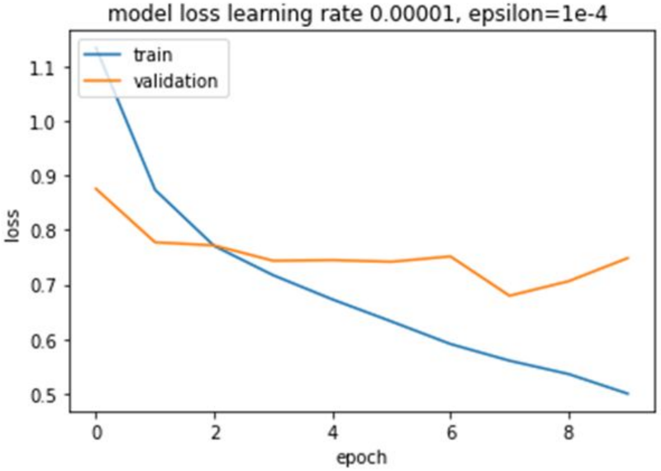
Result(contd.)

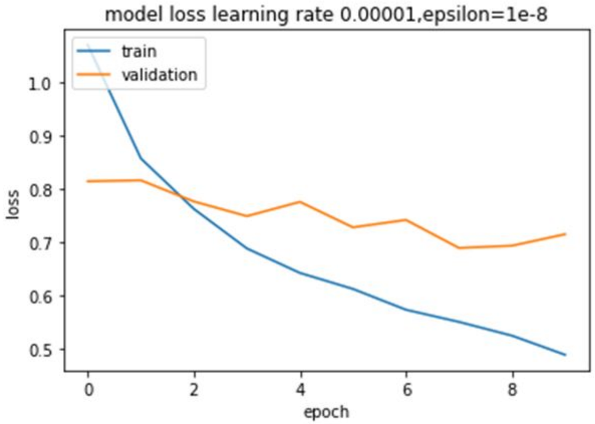
Epsilon

Network: DenseNet121 and CNN





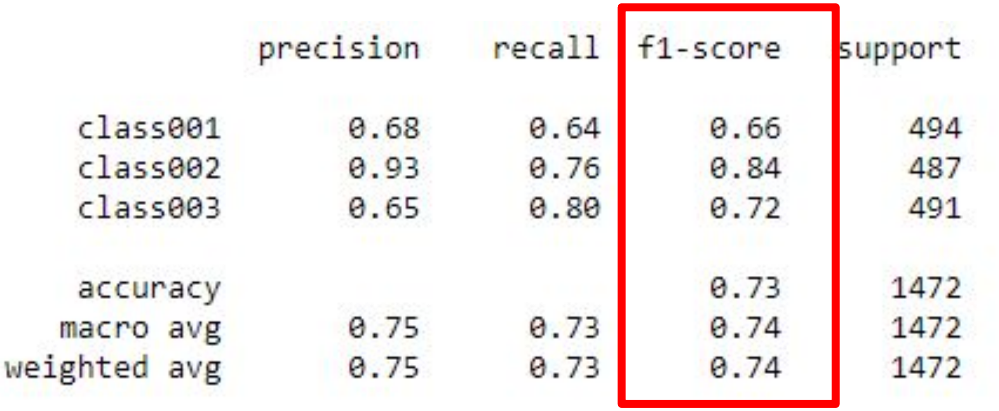




Result(contd.)

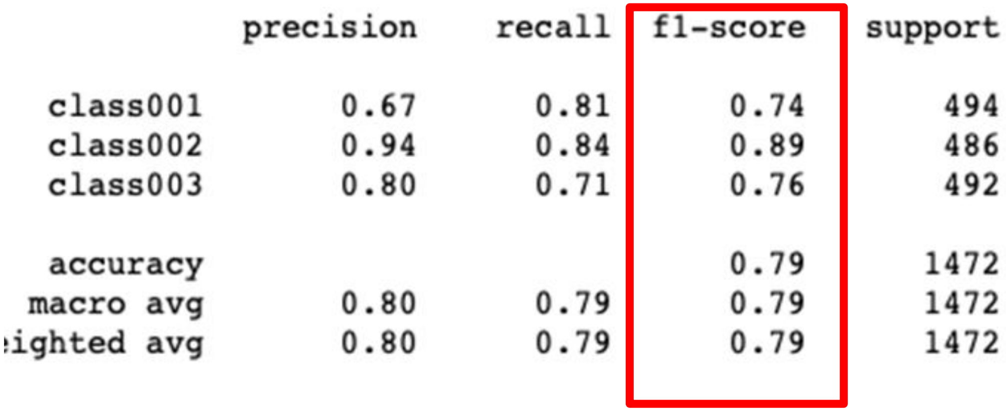
F1 score *using "he_normal"* lr=0.001

Densenet121and CNN



```
              precision    recall  f1-score   support

    class001       0.68      0.64      0.66       494
    class002       0.93      0.76      0.84       487
    class003       0.65      0.80      0.72       491

    accuracy                           0.73      1472
   macro avg       0.75      0.73      0.74      1472
weighted avg       0.75      0.73      0.74      1472
```

VGG16 and CNN



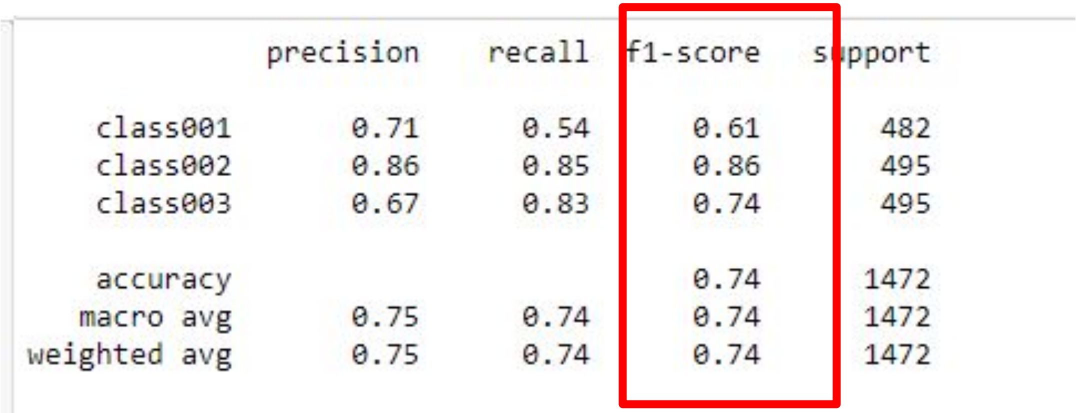
```
          precision    recall  f1-score   support

class001       0.67      0.81      0.74       494
class002       0.94      0.84      0.89       486
class003       0.80      0.71      0.76       492

accuracy                           0.79      1472
macro avg       0.80      0.79      0.79      1472
ighted avg       0.80      0.79      0.79      1472
```

Result(contd.)

F1 score *using* “glorot_uniform

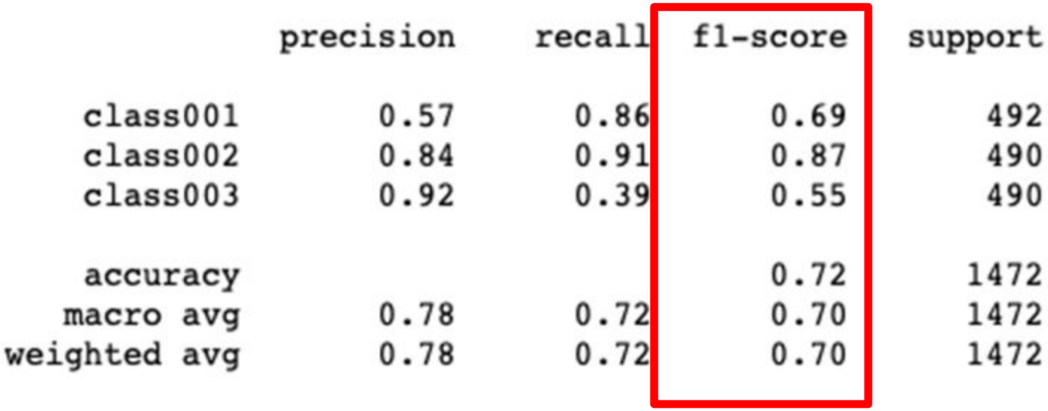
” lr=0.001

Densenet121and CNN



```
              precision    recall  f1-score   support

    class001       0.71      0.54      0.61       482
    class002       0.86      0.85      0.86       495
    class003       0.67      0.83      0.74       495

    accuracy                           0.74      1472
   macro avg       0.75      0.74      0.74      1472
weighted avg       0.75      0.74      0.74      1472
```

VGG16 and CNN



```
              precision    recall  f1-score   support

    class001       0.57      0.86      0.69       492
    class002       0.84      0.91      0.87       490
    class003       0.92      0.39      0.55       490

    accuracy                           0.72      1472
   macro avg       0.78      0.72      0.70      1472
weighted avg       0.78      0.72      0.70      1472
```

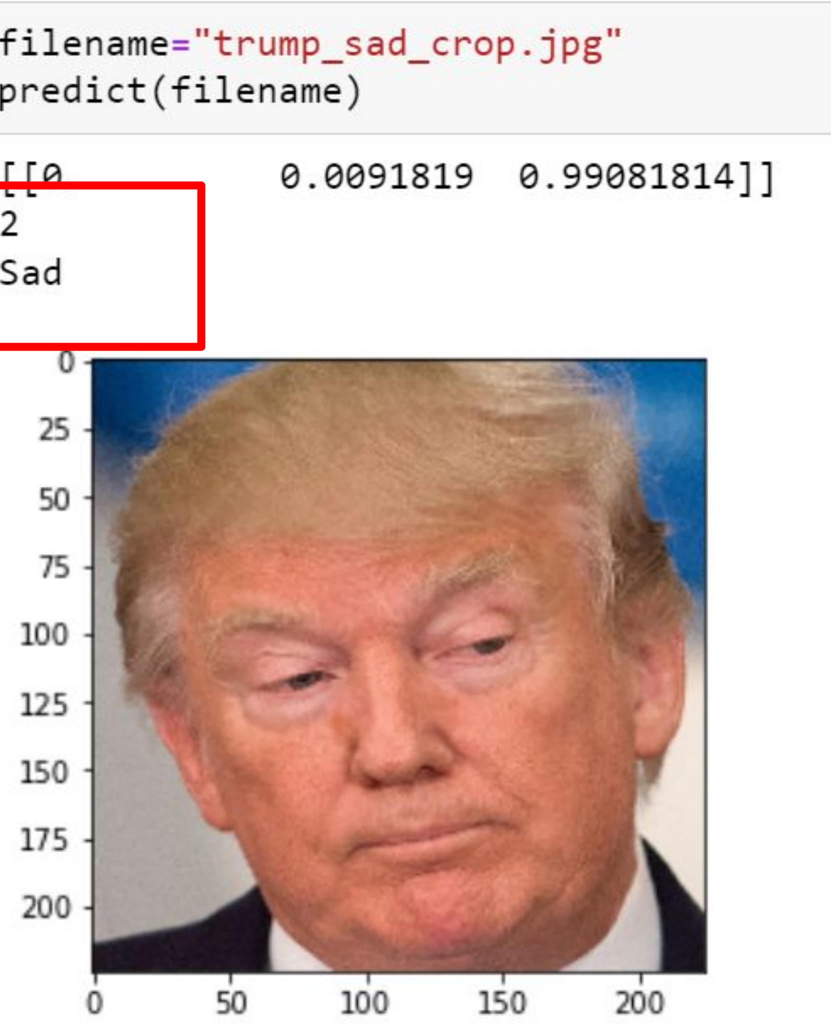
Future work

Combine with Facial Recognition

- Use OpenCv for facial detection

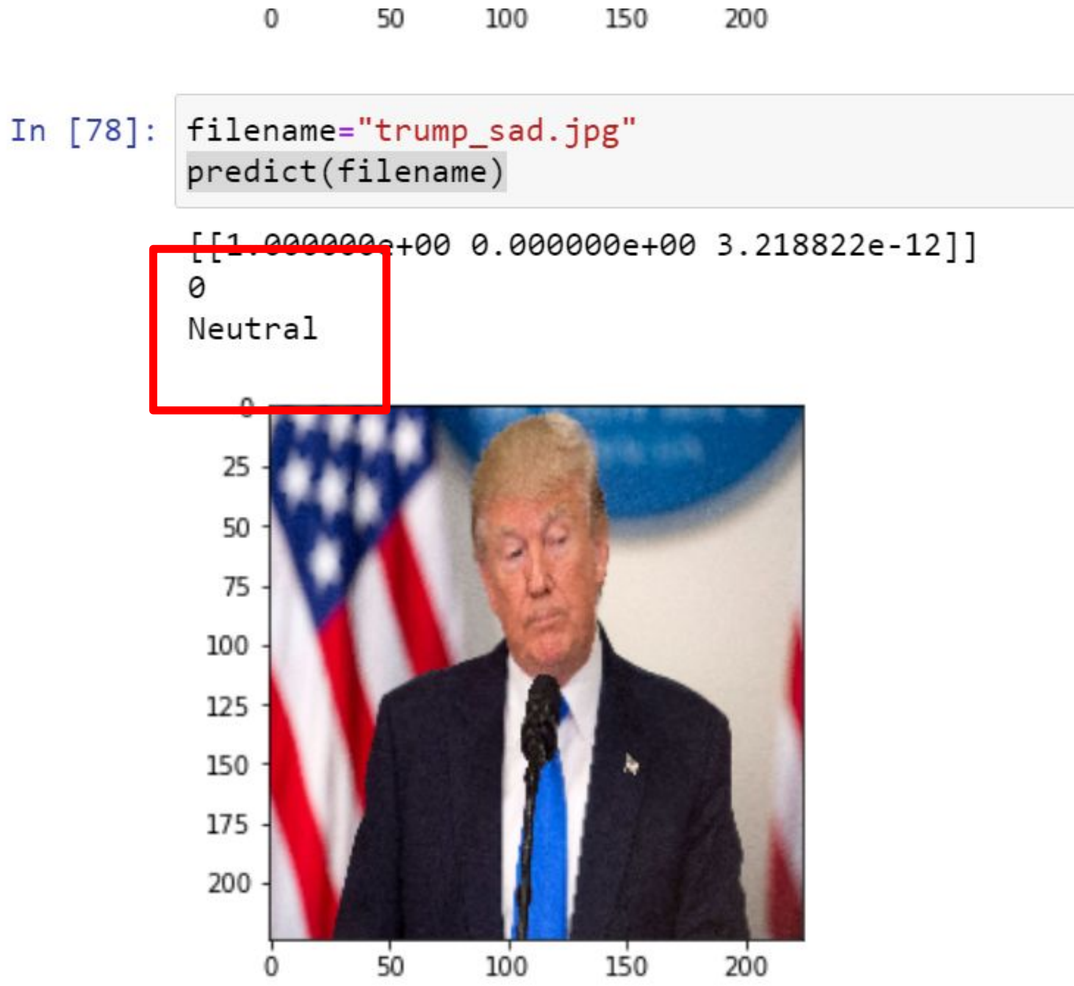
Future work (contd.)

Combine with Facial Recognition



```
filename="trump_sad_crop.jpg"
predict(filename)
```

```
[[0          0.0091819   0.99081814]]
2
Sad
```



```
In [78]: filename="trump_sad.jpg"
         predict(filename)
```

```
[[1.000000e+00 0.000000e+00 3.218822e-12]]
0
Neutral
```

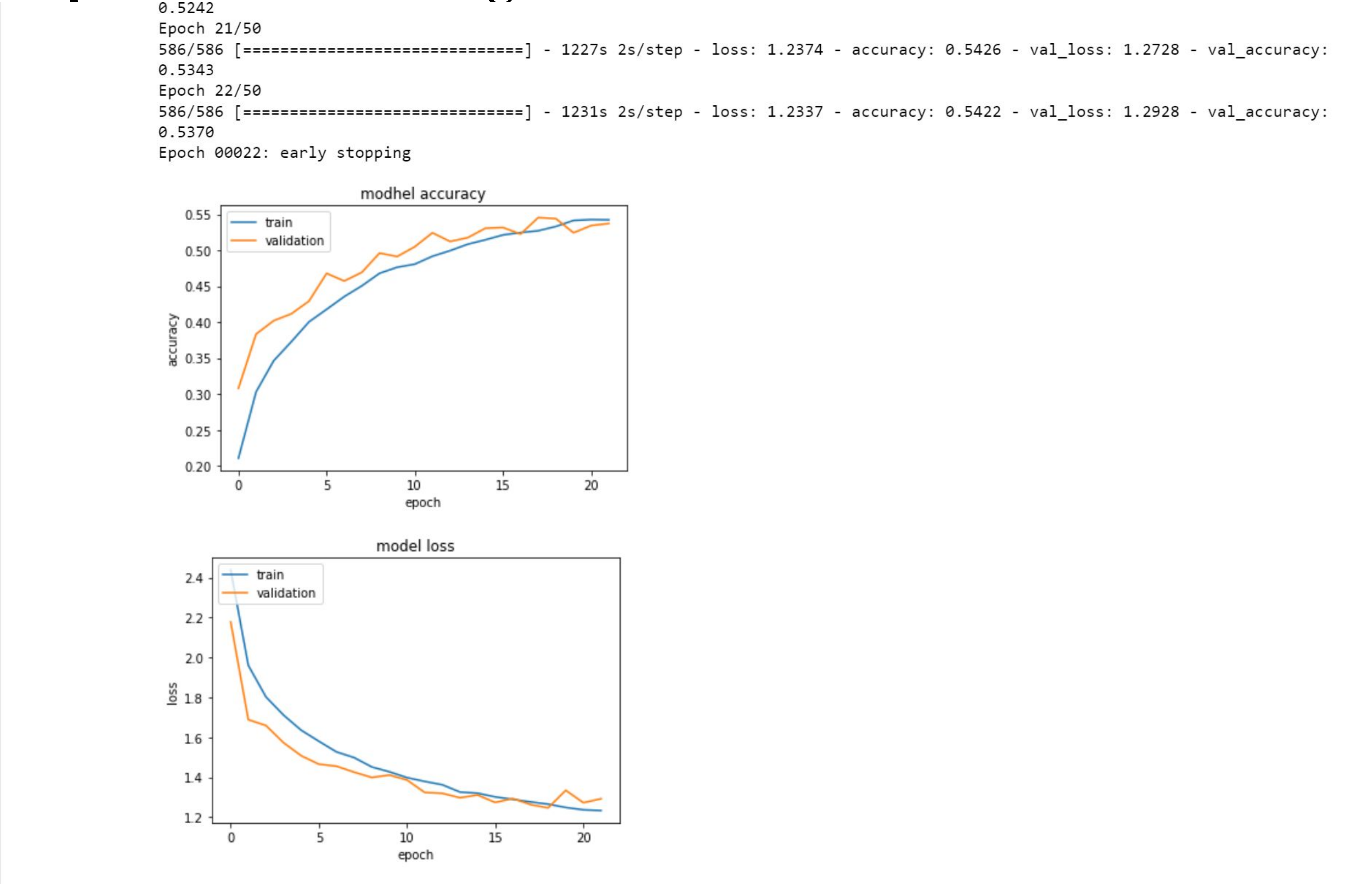
Future work (contd.)

Expand to more categories of emotion

- increase no of classes from 3 to 8
- downside: increase training time (4-6 hour)
  & less accurate (validation accuracy ~50%)

Future work (contd.)

Expand to more categories of emotion



```
0.5242
Epoch 21/50
586/586 [==============================] - 1227s 2s/step - loss: 1.2374 - accuracy: 0.5426 - val_loss: 1.2728 - val_accuracy:
0.5343
Epoch 22/50
586/586 [==============================] - 1231s 2s/step - loss: 1.2337 - accuracy: 0.5422 - val_loss: 1.2928 - val_accuracy:
0.5370
Epoch 00022: early stopping
```

Future work (contd.)

Model with multiple sub branch

- concatenate feature map from more than 1 model(Densenet, resnet, inception) with different combinations.
- cannot achieve within the timeframe of current project

Reference

Reference :

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on ImageNet Classification. *2015 IEEE International Conference on Computer Vision (ICCV)*. https://doi.org/10.1109/iccv.2015.123

Reddi, Sashank J., et al. "On The Convergence Of Adam And Beyond." *ICLR 2018*, 2018, https://doi.org/https://doi.org/10.48550/arXiv.1904.09237.

THE UNIVERSITY OF ALABAMA AT BIRMINGHAM.

Thank you!