We declare that we have completed this assignment completely and entirely on our own, without any consultation with others. We have read the UAB Academic Honor Code and understand that any breach of the Honor Code may result in severe penalties.

We also declare that the following percentage distribution ***faithfully*** represents individual group members' contributions to the completion of the assignment

Name	Overall Contribution (%)	Major work items completed by me	Signature or initials	Date
Parth Patel	25	Data pre	PP	21 April
Rainfield Mak	25		MRF	
Bhadhan Roy Joy	25	Dataset selection, training and tuning of hyper parameters of DenseNet121, result compilation	BRJ	21 April
Pravasini Pati	25	Training models and tuning of hyper parameters	PP	21 April

Dataset link:

https://www.kaggle.com/datasets/mouadriali/affectnetsample

Facial Expression Recognition Using Transfer Learning techniques with VGG16 and DenseNet 121

Abstract

This paper explore the use transfer learning technique to identify 3 different type (Neutral, Happy, Sad) of facial expression. VGG16 and DenseNet121 were used as feature extractor and combine with a CNN with 2 fully connected layer. Weight Initialization , learning rate and epsilon in Adam were chosen to optimize the performance of the Network. Macro F1 score was used to evaluate the model of each Network with the highest validation accuracy. Network using VGG16 and Densenet 121 achieve macro F1 score of 79 and 74 respectively. Future work can be focused on combining existing model with facial recognition model, expand to detect more emotions and Network with multiple subbranch.

Parth Patel,Rainfield Mak, Bhadhan Roy Joy, Pravasini Pati

1. Background/motivation

Human Emotions are the expressions that human present naturally for their situations and circumstances and some of the emotions tends to lead to some actions. There are more than 26 emotions that humans express and some of them are happy, angry, fearful, sad, surprised, etc. These are expressed naturally with a different state of affairs the person is put into.

There have been a lot of convolutional neural network (CNN) models developed in the last few years, some of them include detecting what kind of animal, detecting any objects such as table or fruit from the given image. We wanted to take a step further go which detecting emotions. This model would not only detect the object, in this case, the human face but also the feeling they are expressing. With the hope of a working model, there can be further work done where it can be implemented into an application that could detect human feelings and expressions without even actually interacting with them. Moreover, it could be used by the companies to obtain their customer satisfaction score through emotions only and not through a survey. Furthermore, this idea would be a step further in the world of Artificial Intelligence (AI) if the machine learns and could identify human emotion(Uppal et.al, 2019).

2.Proposed Method

2.1.Model & Network Structure . In this project , VGG 16 and DenseNet121 were used. The two models were chosen due to the difference in the no of layers. The complete network consist of DenseNet121 and CNN, same for VGG16. Both models were used as feature extractor and each connect to the same CNN. The CNN consisted of a flatten layer to flatten the outputs from VGG16 and DenseNet121. The CNN consist of two fully connected (fc) layers .The final layer act as classifier. The first fc layer using "Relu" and "He_intializer" to reduce the overfitting problem(He et al., 2015).The classifier use "Softmax" as the activation function. In between the two fc layers , there is one layer of batch normalization followed by a layer of drop out. The detail structure of the CNN can be shown in Figure 1.

```
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
_________________________________________________________________
```

Figure 1. Model summary of sequential model

2.2.Data preprocessing. To avoid overfitting, the training data was under data augmentation before being pass to the DenseNet121 and VGG16. The data augmentation process include angular rotation, rescaling and horizonal flip.

2.3.Network Training. The fundamental concept of network training was to use both the Densenet121 and VGG16 as feature extractor. The output from both of model were then passed to the CNN model for training.All the training was set to be 10 epochs due to the limited amount of time(each training time would be around 70 minutes). Due to the limited amount of time, only three hyperparameter i) "kernel_intializer", ii) "learning rate" iii) "epsilon" in the Adam optimizer.

i. For kernel initializer, since "Relu" was chosen as the activation function,"he_normal" in theory would be the appropriate option for weight initialization when comparing to "Xavier" (He et al., 2015). Both networks were first trained without the any weight initialization .The same process would then repeat again with "he_normal" as the kernel initializer.

ii. As learning rate would always had a huge impact on the overall accuracy of the model. Both model would be train with the same set of learning rate values(1e-3,1e-4,1e-5) to compare the performance.

iii. The default "epsilon" in the Adam optimizer may not be the optimal setting which caused Adam not able to converge to optimal solution(Reddi et.al, 2018). Both network would then train with the same set of epsilon values (1e-8,1e-4,0.1,1) to compare the performance.

3.Result

3.1.Weight initialization. The training accuracy and validation accuracy for both network using "he_normal" both increased when compared to using "glorot_uniform". The increase was more notable in network using VGG16. Detail result was shown in Table 1.

Kernel Initializer /lr=1e-2	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

Table1. Training accuracy and validation accuracy on weight initialization.

3.2.Learning rate. 3 different learning rate were chosen to optimize the performance of both network. Network using VGG16 has the highest training accuracy when learning rate equal to 1e-4. For DenseNet121,it would be when learning rate equal to 1e-3 which is the same for Validation Accuracy. Detail result was shown in Table 2.

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.

3.3.Epsilon. 5 different epsilon values (include the default one) were chosen to fine tune the Adam optimizer. For Network with VGG16, it reached the highest validation accuracy when epsilon equal 1. For Network with DenseNet121, highest accuracy was reached when epsilon equal to 1e-4.

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

Table3. Training accuracy and validation accuracy on different epsilon value.

3.4.F1 score. The best (highest validation accuracy) model of each Network was chosen to calculate the macro average F1 score. The macro average F1 for Network using VGG16 was 0.74 while the Network using DenseNet121 achieve a score of 74. Detail score was show in Table 4 and Table 5.

	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

Table 4. F1 score of Network using DenseNet121, lr=0.001 , kernel_intializer="he_normal".

	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
weighted avg	0.80	0.79	0.79	1472

Table 5. F1 score of Network using DenseNet121, lr=0.001 , kernel_intializer="he_normal".

4.Discussion

For the optimization on the on weight initialization method , bot Network have an increase in accuracy in when using the "he_normal" which coincided with the idea proposed by He.,K in 2015 (He et al., 2015). For the tunning in epsilon value, the result for Network demonstrate that the default value provide by tensor in the Adam optimizer might not always be the optimal choice. For Network using VGG16 , the Network reach the highest validation accuracy when epsilon equal to 1. In this project , the set of epsilons values being used were only train on both Network with the same learning rate (lr =1e-5). In future work , a combination of different learning rate and epsilon values should be used to find the optimal combination of the Network being work on.

5.Future Work.

5.1Combine with Face Recognition. Current model only expected to receive image mostly comprise of human face and did not support facial recognition. When non-human face object such as background and body were being introduced, it would greatly reduce the performance of the model. OpenCV provide existing facial detection function that can be used to crop out the necessary part to feed to the model.

5.2Expand to more categories of emotion. Current model only detect 3 emotions(neutral , sad, happy) with moderate accuracy. There is existing dataset that have 8 classes of emotions, however, the entire set consist of 30k image which would take around 3-4 hours for each training, which is not optimal with the given time frame. Thus future work could be done to address the issue.

5.3Model with multiple subbranch. Current model only consist of single branch, model with multiple sub branches such as merging the feature of VGG16 and Densent121 maybe able to generate better results, however, given the time frame of this project we only manage to train it briefly with beyond optimal accuracy

Reference

A. Uppal, S. Tyagi, R. Kumar and S. Sharma, "Emotion recognition and drowsiness detection using Python," 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2019, pp. 464-469, doi: 10.1109/CONFLUENCE.2019.8776617.

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.