A blue and black logo

Description automatically generated

**Abdelrahman Hassan Ismail  
2201662**

**CSE616 (PG2015) - Neural Networks and Their Applications**

Under supervision: Dr.Mahmoud Khalil

**Abstract**

The aim of this research project is to construct an image classification model that can effectively differentiate between images depicting happy individuals and those that do not. An additional objective is to assess the extent to which the model captures features associated with happiness, both in terms of facial expression recognition and broader contextual cues. Following training, testing, and validation processes, the model will be manually evaluated using a diverse set of images, encompassing various subjects such as happy emojis, animals expressing happiness, and contrasting images portraying sadness.

Dataset

o <https://www.kaggle.com/datasets/jessicali9530/celeba-dataset>

o Size: 1.5GB

o No. of Photos: 200K Real Celebrities Faces

o Attributes: 41 (Include Smiling attribute)

o No. Of Unique Identities: 10K

Transfer Learning ResNet50 only train last 5 layers

Model Architecture: The model utilized in this study is a sequential neural network architecture, composed of various layers. The initial layer is a pre-trained ResNet50V2 model, distinguished by its capacity for deep feature extraction.

This layer outputs feature maps with dimensions of 8x8x2048, having a total of 23,564,800 parameters. Following the ResNet50V2 layer, a Global Average Pooling 2D layer is employed, resulting in an output shape of 2048. This layer serves the purpose of extracting global features from the previous layer. Subsequently, a series of dense layers are incorporated into the model.

The first dense layer contains 512 neurons, the second dense layer contains 256 neurons, the third dense layer contains 128 neurons, and the fourth and final dense layer contains 1 neuron.

The total number of parameters for each dense layer are as follows: 1,049,088 for the first layer, 131,328 for the second layer, 32,896 for the third layer, and 129 for the fourth layer. Overall, the model encompasses a total of 24,778,241 parameters, out of which 2,268,161 are trainable, while the remaining 22,510,080 are non-trainable.

Dataset Specifications: The dataset used for training, validation, and testing consists of a total of 114,000 images, all of which are cropped to focus solely on the facial region. Out of these, 14,000 images are allocated for validation purposes, while the remaining 14,000 images are reserved for testing the model's performance.

A graph with orange lines and numbers

Description automatically generatedA graph with orange and blue lines

Description automatically generated

**Conclusion:** i get from the two curves that there are overfitting happening so I decided to and the **testing accuracy was 0.84**

# Transfer learning ResNet50V2 only train last layer

# Model Architecture:

# The sequential neural network employed in this experiment is referred to as "sequential\_9." It comprises several layers, starting with a pretrained ResNet50V2 model. The ResNet50V2 layer outputs feature maps with dimensions of 8x8x2048. This layer has a total of 23,564,800 parameters.

# After the ResNet50V2 layer, a Global Average Pooling 2D layer is introduced, resulting in an output shape of 2048. This layer extracts global features from the preceding layer.

# Following the pooling layer, two dense layers are incorporated into the model. The first dense layer consists of 128 neurons, and the second dense layer consists of 1 neuron. The first dense layer has 262,272 parameters, while the second dense layer has 129 parameters.

# In total, the model contains 23,827,201 parameters, out of which 262,401 are trainable, and the remaining 23,564,800 are non-trainable.

# Dataset Specifications:

# The dataset utilized for this experiment includes 32,000 images, which have all been cropped to focus exclusively on the facial region. Out of these, 6,400 images are allocated for validation, and 3,200 images are used for testing the model's performance.

# Experimental Details:

## To expedite the experimental process, the model was trained for 20 epochs on the reduced dataset size mentioned above. Only the last dense layer and the ResNet50V2 layers were trained, while the remaining layers were frozen

## A graph with orange and blue lines Description automatically generatedA graph with blue and orange lines Description automatically generated

## Conclusion:

## I get that the two curves the model acting good and I wanted to try to increase number of epochs to see what I will get from that model Testing accuracy is 0.795625

# Transfer learning ResNet50V2 only train last layer increase epochs to 100

# Model Architecture:

# The same as last model but increase number of epochs.

## A graph with orange lines and blue lines Description automatically generated

## 

## Conclusion:

## It shows a true overfitting but with acceptable accuracy somehow for me testing accuracy is 0.8684375

# Transfer learning EfficenetV2B only train 5 last layers and only last layer

# Model Architecture:

5 last layers training model:

* Total params: 6,083,409

Trainable params: 164,097

Non-trainable params: 5,919,312

Last layer training model:

* Total params: 6,083,409

Trainable params: 164,097

Non-trainable params: 5,919,312

20 Epochs Each  
with training data 32K Images and validation data 3.2K and testing data 3.2K  
  
the model stuck at 0.51 through whole training

## Conclusion:

## It shows that the model is not that good for fine tuning

# Transfer learning EfficenetV2B0 Train the Whole Model

# Model Architecture:

* Input: The model expects input images of size 224x224 pixels with three color channels (RGB).
* Convolutional Layers: The model begins with a standard convolutional layer, which is followed by a series of repeated blocks. Each block consists of a depthwise separable convolution, batch normalization, and a nonlinear activation function (such as the Swish function).
* Depthwise Separable Convolution: This type of convolution separates the spatial and channel-wise operations, reducing the number of parameters and computational cost. It consists of two steps: depthwise convolution, which applies a single convolutional filter per input channel, and pointwise convolution, which applies a 1x1 convolutional filter to combine the output channels.
* Blocks: The EfficientNet-B0 model contains a total of 7 blocks. The number of repetitions and the number of filters in each block vary, gradually increasing as the network goes deeper. These blocks help capture features at different scales and complexities.
* Global Average Pooling: After the convolutional layers, the model applies global average pooling to reduce the spatial dimensions of the feature maps to a single vector. This operation aggregates information from the entire feature map.
* Fully Connected Layers: The global average pooled features are then passed through a fully connected layer, followed by a softmax activation function, which produces the final class probabilities. The number of neurons in the fully connected layer corresponds to the number of target classes

20 Epochs   
with training data 32K Images and validation data 3.2K and testing data 3.2K

Total params: 6,083,409

Trainable params: 6,022,801

Non-trainable params: 60,608

A graph with orange and blue lines

Description automatically generatedA graph of loss and loss

Description automatically generated

## Conclusion:

## It shows a lot of fluctuations and also overfitting appears clearly I thought that the model is somehow training to figure very hard feature so I tried to increase epochs of training testing results : 0.6984375 which is the lowest

# Transfer learning EfficenetV2B0 Train the Whole Model 50 epochs

# Model Architecture:

Same last Arch.

50 Epochs   
with training data 32K Images and validation data 3.2K and testing data 3.2K

Total params: 6,083,409

Trainable params: 6,022,801

Non-trainable params: 60,608

A graph with orange lines

Description automatically generated

A graph with orange lines

Description automatically generated

## Conclusion:

## Increasing epochs just show the fluctuating so clear and the overfitting appear more clear testing accuracy is 0.89625

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model/Info | Base Model | Transfer Learning | Epochs | Training Data (Images) | Validation Data (Images) | Testing Data (Images) | Learnable parameters | Non-Learnable parameters |
| Model 1 | Resnet50V2 | 5 layers | 20 | 114K | 14K | 14K | 2.2M | 24.7M |
| Model 2 | Resnet50V2 | Last layer only | 20 | 32K | 6.4K | 3.2K | 0.262M | 24M |
| Model 3 | Resnet50V2 | Last layer only | 100 | 32K | 6.4K | 3.2K | 0.262M | 24M |
| Model 4 | EfficentNetV2B0 | All Layers | 20 | 32K | 6.4K | 3.2K | 6M | 0.06M |

***Note: I excluded fine tuning model of EfficentNetV2B0 due its poor performance***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model/Info | Model 1 | Model 2 | Model 3 | Model 4 |
| Base Model | Resnet50V2 | Resnet50V2 | Resnet50V2 | EfficentNetV2B0 |
| Transfer Learning | 5 layers | Last layer only | Last layer only | All Layers |
| Epochs | 20 | 20 | 100 | 20 |
| Training Data (Images) | 114K | 32K | 32K | 32K |
| Validation Data (Images) | 14K | 6.4K | 6.4K | 6.4K |
| Testing Data (Images) | 14K | 3.2K | 3.2K | 3.2K |
| Learnable parameters | 2.2M | 0.262M | 0.262M | 6M |
| Non-Learnable parameters | 24.7M | 24M | 24M | 0.06M |
| Val\_Acc/Acc |  | A graph with orange and blue lines  Description automatically generated |  | A graph with orange lines  Description automatically generated |
| Val\_Loss vs Loss |  |  |  |  |
| Testing Data Accuracy | 0.8492853 | 0.795625 | 0.8684375 | 0.6984375 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model/Manual Testing | Model 1 | Model 2 | Model 3 | Model 4 |
|  | 100% Correct | 87% Correct | 100% Correct | 100% Correct |
| A close up of a person's face  Description automatically generated | 100% Correct | 100% Correct | 100% Correct | 100% Correct |
|  | 100% Correct | 100% Correct | 100% Correct | 100% Correct |
| A person smiling with his mouth open  Description automatically generated | Wrong 100% | 100% Wrong | 100% Correct | 100% Correct |
| A close-up of a person smiling  Description automatically generated | Wrong 100% | Wrong 100% | 100% Correct | 100% Correct |
| A close up of a person  Description automatically generated | 100% Correct | 100% Correct | 55% only Right the model predicted it as smiling and somehow iam with that percentage as it hard to tell | 100% Correct |
|  | 100% Correct | Wrong 100% | 100% correct prediction not smiling | 82% right predicting not smlling with 82% |
|  | Wrong 100% | 100% Correct | 100% correct predict smiling | 100% Correct |
|  | 100% Correct | Wrong 100% | Wrong 100%  predect smiling | Wrong 100% |
|  | 100% Correct | Wrong 100% | 100% Correct | 100% Correct |
|  | Wrong 100% | 100% Correct | Wrong 100% | Wrong 100% |
|  | Wrong 100% | 100% Correct | 100% Correct | Wrong 100% |

**Conclusion**

**Model 3 Gives very satisfying results through its auto testing with 3.2K images also with manual testing with weird data   
  
I tried to see if the model capture the smiling feature instead of smiling human and its somehow been predicted it right   
  
so fine tuning is really better than training a whole model from scratch   
  
and the more important is the data   
  
as we saw that training a a lot of layers as effinectNetV2B0 and train the whole model itself doesn’t gives a good results compared to fine tune a pre trained model have seen a lot a lot of data as resnet50v2**

References

To develop the image classifier, the following key references will be useful: • Tan, M. & Le, Q.. (2021). EfficientNetV2: Smaller Models and Faster Training. • Proceedings of the 38th International Conference on Machine Learning, in • Proceedings of Machine Learning Research 139:10096-10106 Available from • <https://proceedings.mlr.press/v139/tan21a.html>   
  
"Identity Mappings in Deep Residual Networks" by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun.

<https://arxiv.org/pdf/1603.05027.pdf>