Deep Learning – Case Study

Facial Expression Detection Using CNN

Name: Jay Patel

Enrollment Number: 19012012004

Batch: DL2

1. Introduction

This case study is designed to Classify Facial Expressions using images. It uses the Sequential model for the images. The model built is able to grid the image and train the model accordingly and then predict suitable expressions from them.

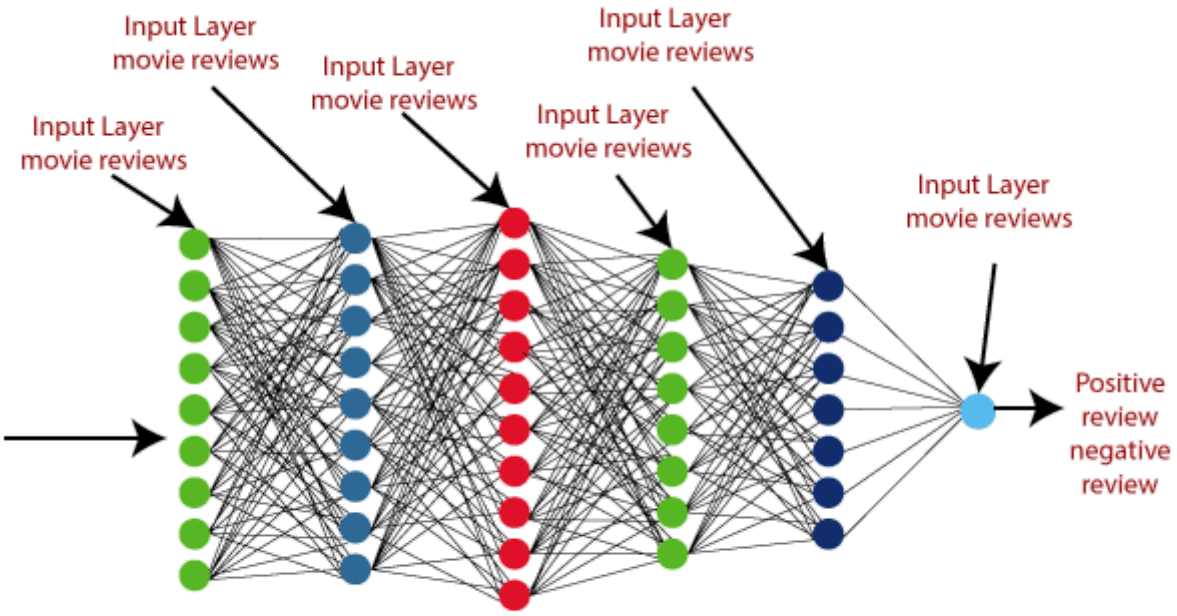
This case study can be used across various sectors where there is a need to have the Facial Expression classification.

2. Tools and Technologies

Tools and Libraries	Usage
Keras	This library is used for building the network architecture. It allows us to use several layers, callbacks, and the InceptionResNetV2 model.
Sequential	A Sequential model is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor.
matplotlib	Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy.
Kaggle	Used as a platform to execute code as well as manage datasets and model weights

3. Model Explanation and Architecture

In a sequential model, we stack layers sequentially. So, each layer has unique input and output, and those inputs and outputs then also come with a unique input shape and output shape. If you want to retrieve the weights a layer has, you can simply call get weights on the layer and retrieve all those weights as a list of NumPy arrays. First, we instantiate a sequential model, then we add layers to it one by one. Afterwards, we need to compile a model with a mandatory loss function and a mandatory optimizer, and if we like, also with optional evaluation metrics such as accuracy.



Keras Neural Network Sequential Model

4. Working

The sequential model is used with three convolutional layers and two dense layers. Each convolutional layer has batch normalization, 2x2 max pooling and 0.2 dropouts. The activation function used in convolutional layers is relu. The Dense layer has 512 nodes and relu activation. It also has batch normalization and 0.5 dropouts. The output layer has two output nodes and a softmax activation function. The model uses categorical cross-entropy for loss and red prop optimizer.

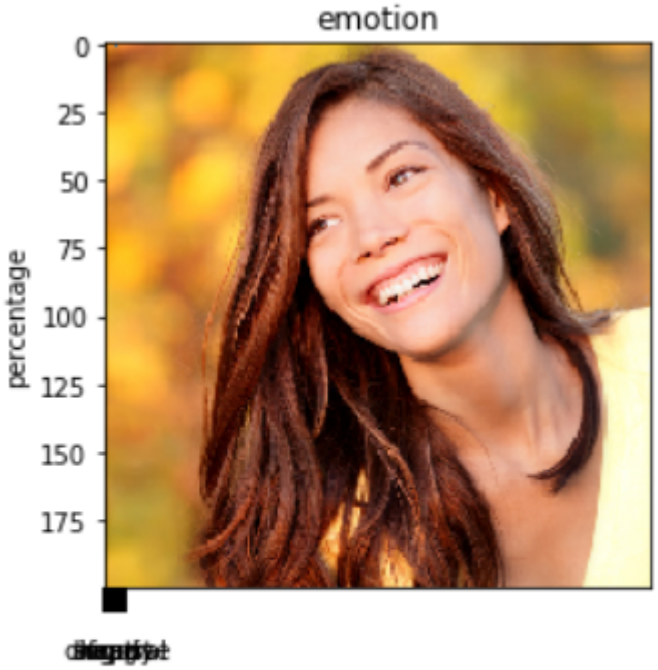
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 48, 48, 64)        1664
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 48, 48, 64)        102464
_________________________________________________________________
batch_normalization_1 (Batch (None, 48, 48, 64)        256
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 24, 24, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 24, 24, 128)       204928
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 24, 24, 128)       409728
_________________________________________________________________
batch_normalization_2 (Batch (None, 24, 24, 128)       512
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 12, 12, 128)       0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 12, 12, 256)       295168
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 12, 12, 256)       590080
_________________________________________________________________
batch_normalization_3 (Batch (None, 12, 12, 256)       1024
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 6, 6, 256)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 9216)              0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               1179776
_________________________________________________________________
batch_normalization_4 (Batch (None, 128)               512
_________________________________________________________________
activation_1 (Activation)    (None, 128)               0
_________________________________________________________________
dropout_1 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 7)                 903
_________________________________________________________________
activation_2 (Activation)    (None, 7)                 0
=================================================================
Total params: 2,787,015
Trainable params: 2,785,863
Non-trainable params: 1,152
_________________________________________________________________
```

5. Code

https://github.com/jaypja99/Facial-Expression-Detection-Using-CNN/blob/main/facial-expression-detection-cnn%20(1).ipynb

** Code and Report are available in the GitHub repository.

6. Output




```
Expression Prediction: happy
```
