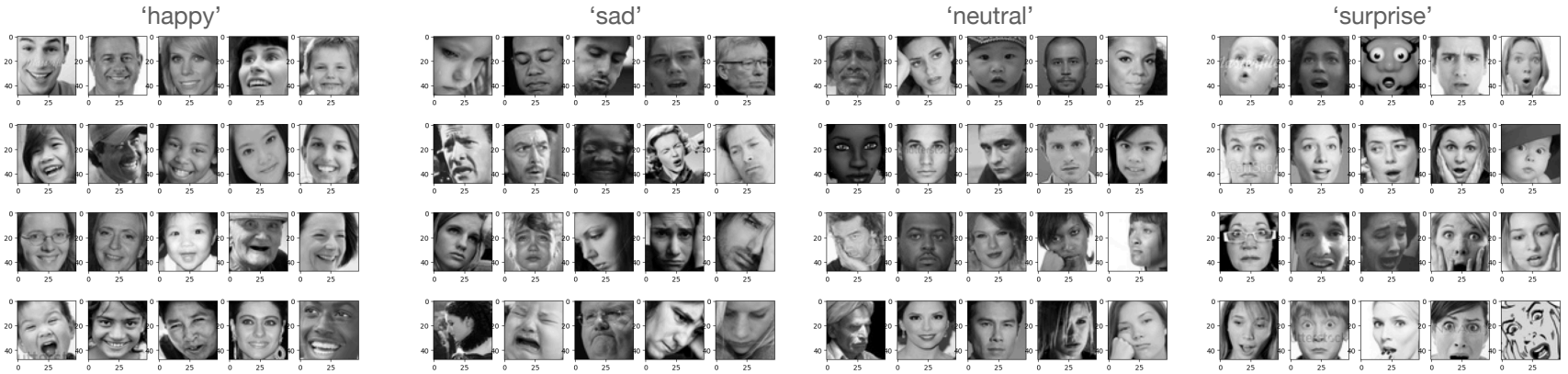
Facial Emotion Recognition

Convolutional Neural Networks, Tensorflow, Keras, Transfer Learning (VGG16, ResNet)

‘Femi Bolarinwa
f3bolarinwa@yahoo.com

Data Snapshot

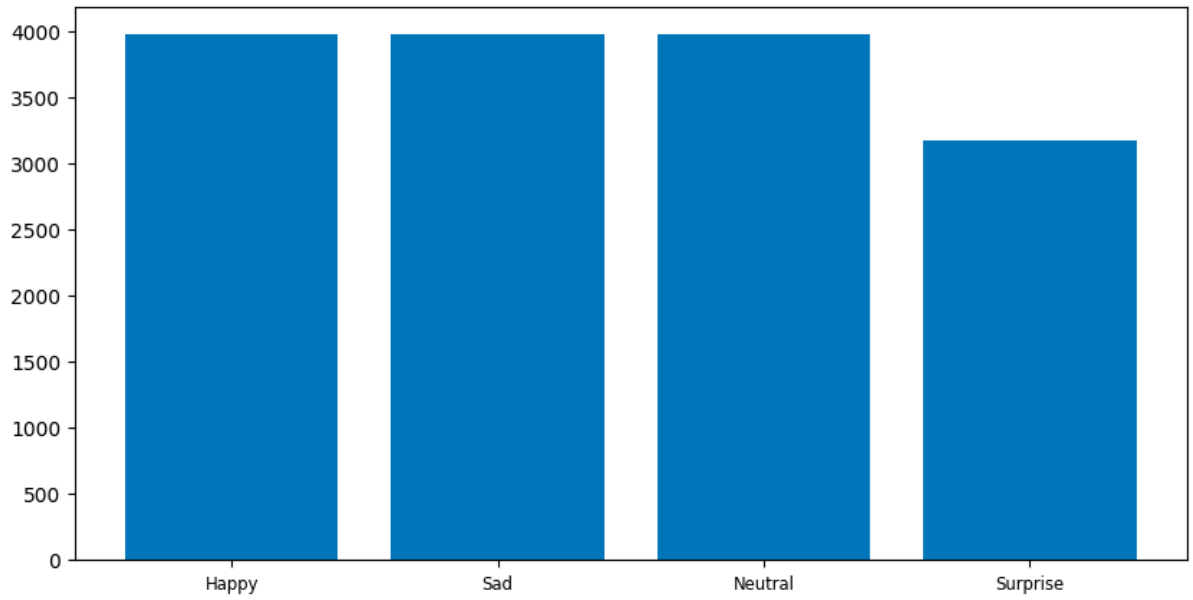
Classes of Facial Emotions in Dataset



- Dataset has 3 folders: ‘train’, ‘validation’ and ‘test’
- Each folder consist 4 classes of images: ‘happy’, ‘sad’, ‘surprise’ and ‘neutral’
- Images converted to 48x48 pixels using python’s ImageDataGenerator
- One-hot encoding for class labels generated by ImageDataGenerator
- Pixels normalized to prevent exploding gradient

Distribution of Classes in Training Data

- Evenly distributed classes except ‘surprise’
- Imbalance not significant enough to cause bias away from ‘surprise’



CNN Model 1

Architecture

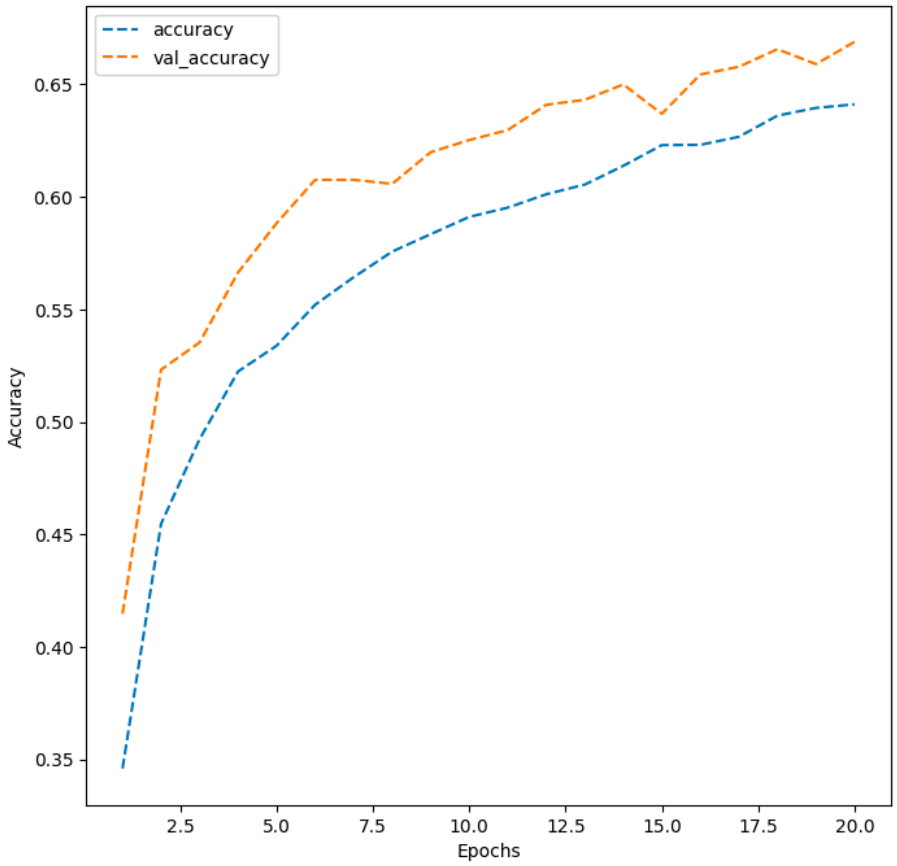
- 3 convolutional blocks
- Single channel input (‘grayscale’ colour mode)
- Maxpooling, ‘same’ padding, ‘relu’ activation
- Dropout layers for regularization
- 1 extra dense layer, ‘relu’ activation (fully connected layer)
- ‘softmax’ activation for multi-class classification at output layer
- 605,060 trainable parameters (neuron weights and biases)

```
model1.summary()

Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 48, 48, 64)        320

 max_pooling2d_3 (MaxPooling  (None, 24, 24, 64)       0
 2D)

 dropout_3 (Dropout)         (None, 24, 24, 64)        0

 conv2d_4 (Conv2D)           (None, 24, 24, 32)        8224

 max_pooling2d_4 (MaxPooling  (None, 12, 12, 32)       0
 2D)

 dropout_4 (Dropout)         (None, 12, 12, 32)        0

 conv2d_5 (Conv2D)           (None, 12, 12, 32)        4128

 max_pooling2d_5 (MaxPooling  (None, 6, 6, 32)         0
 2D)

 dropout_5 (Dropout)         (None, 6, 6, 32)          0

 flatten_1 (Flatten)         (None, 1152)              0

 dense_1 (Dense)             (None, 512)               590336

 dropout_6 (Dropout)         (None, 512)               0

 dense_2 (Dense)             (None, 4)                 2052

=================================================================
Total params: 605,060
Trainable params: 605,060
Non-trainable params: 0
```

CNN Model 1

Training and Evaluation

- 20 epochs
- ‘Adam’ optimizer, learning rate = 0.001
- ‘categorical-crossentropy’ loss function, ‘accuracy’ metric
- ~64% accuracy on training, validation and unseen test data
- Generalized performance, not overfitted
- Model can achieve better accuracy if run for longer epoch



CNN Model 2

Architecture

- 4 convolutional blocks
- Single channel input (‘grayscale’ colour mode)
- Maxpooling, ‘same’ padding, ‘relu’ activation
- BatchNormalization and Dropout layers for regularization
- 1 extra dense layers, ‘relu’ activation (fully connected layer)
- ‘softmax’ activation for multi-class classification at output layer
- 388,644 trainable parameters (neuron weights and biases)

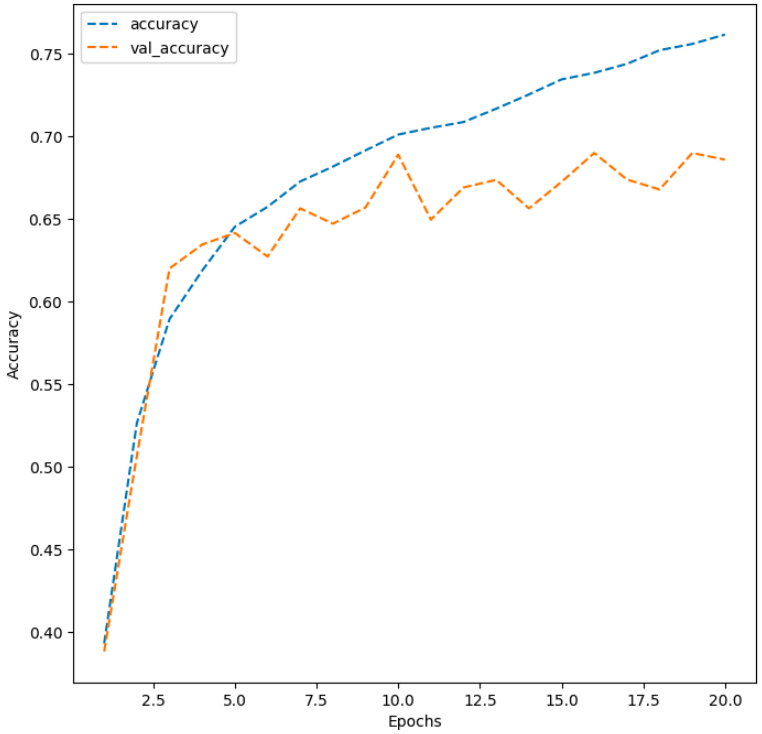
```
model2.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_10 (Conv2D)          (None, 48, 48, 256)       1280

 batch_normalization_14 (Bat  (None, 48, 48, 256)      1024
 chNormalization)

 leaky_re_lu_14 (LeakyReLU)  (None, 48, 48, 256)       0

 max_pooling2d_10 (MaxPoolin  (None, 24, 24, 256)      0
 g2D)

 conv2d_11 (Conv2D)          (None, 24, 24, 128)       131200

 batch_normalization_15 (Bat  (None, 24, 24, 128)      512
 chNormalization)

 leaky_re_lu_15 (LeakyReLU)  (None, 24, 24, 128)       0

 max_pooling2d_11 (MaxPoolin  (None, 12, 12, 128)      0
 g2D)

 conv2d_12 (Conv2D)          (None, 12, 12, 64)        32832

 batch_normalization_16 (Bat  (None, 12, 12, 64)       256
 chNormalization)

 leaky_re_lu_16 (LeakyReLU)  (None, 12, 12, 64)        0

 max_pooling2d_12 (MaxPoolin  (None, 6, 6, 64)         0
 g2D)

 conv2d_13 (Conv2D)          (None, 6, 6, 32)          8224

 batch_normalization_17 (Bat  (None, 6, 6, 32)         128
 chNormalization)

 leaky_re_lu_17 (LeakyReLU)  (None, 6, 6, 32)          0

 max_pooling2d_13 (MaxPoolin  (None, 3, 3, 32)         0
 g2D)

 flatten_2 (Flatten)         (None, 288)               0

 dense_6 (Dense)             (None, 512)               147968

 dense_7 (Dense)             (None, 128)               65664

 dense_8 (Dense)             (None, 4)                 516

=================================================================
Total params: 389,604
Trainable params: 388,644
Non-trainable params: 960
_________________________________________________________________
```

CNN Model 2

Training and Evaluation

- 20 epochs
- ‘Adam’ optimizer, learning rate = 0.001
- ‘categorical-crossentropy’ loss function, ‘accuracy’ metric
- >72% accuracy on training and unseen test data
- Generalized performance, not overfitted
- Divergent training and validation accuracies. Model could become overfitted if trained for longer epoch.



CNN Model 3

Architecture (Transfer Learning)

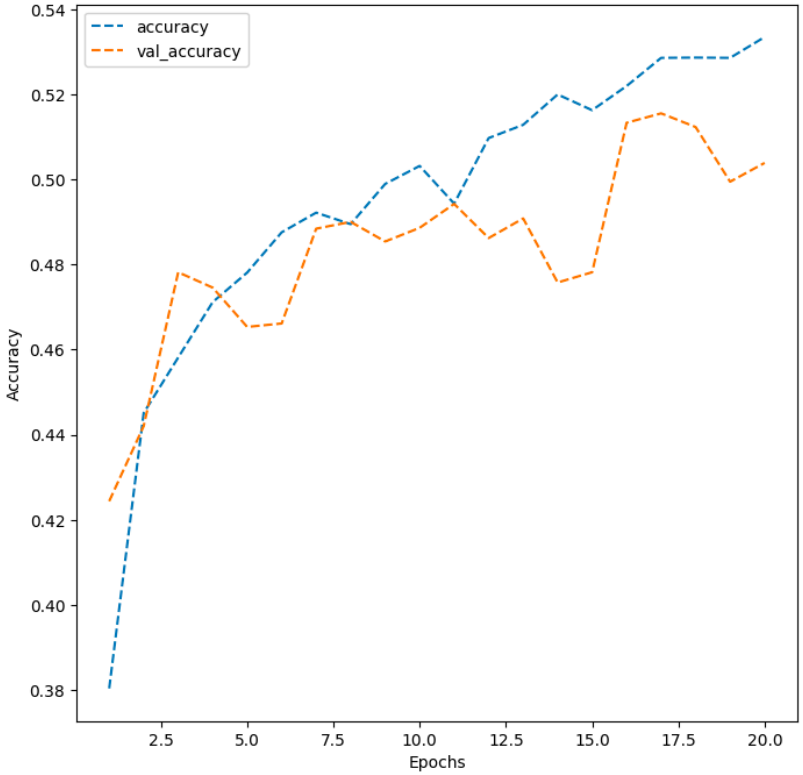
- **VGG16** convolutional and pooling layers imported
- Set transferred VGG16 parameters untrainable
- 3 channel input (‘rgb’ colour mode)
- 3 extra dense layers, ‘relu’ activation (fully connected layers)
- BatchNormalization and Dropout layers for regularization
- ‘softmax’ activation for multi-class classification at output layer
- 172,868 trainable parameters
- 14,714,816 untrainable parameters

```
vggmodel.summary()
block1_conv1 (Conv2D)       (None, 48, 48, 64)        1792
block1_conv2 (Conv2D)       (None, 48, 48, 64)        36928
block1_pool (MaxPooling2D)  (None, 24, 24, 64)        0
block2_conv1 (Conv2D)       (None, 24, 24, 128)       73856
block2_conv2 (Conv2D)       (None, 24, 24, 128)       147584
block2_pool (MaxPooling2D)  (None, 12, 12, 128)       0
block3_conv1 (Conv2D)       (None, 12, 12, 256)       295168
block3_conv2 (Conv2D)       (None, 12, 12, 256)       590080
block3_conv3 (Conv2D)       (None, 12, 12, 256)       590080
block3_pool (MaxPooling2D)  (None, 6, 6, 256)         0
block4_conv1 (Conv2D)       (None, 6, 6, 512)         1180160
block4_conv2 (Conv2D)       (None, 6, 6, 512)         2359808
block4_conv3 (Conv2D)       (None, 6, 6, 512)         2359808
block4_pool (MaxPooling2D)  (None, 3, 3, 512)         0
block5_conv1 (Conv2D)       (None, 3, 3, 512)         2359808
block5_conv2 (Conv2D)       (None, 3, 3, 512)         2359808
block5_conv3 (Conv2D)       (None, 3, 3, 512)         2359808
block5_pool (MaxPooling2D)  (None, 1, 1, 512)         0
flatten_3 (Flatten)         (None, 512)               0
dense_9 (Dense)             (None, 256)               131328
dense_10 (Dense)            (None, 128)               32896
dropout_14 (Dropout)        (None, 128)               0
dense_11 (Dense)            (None, 64)                8256
batch_normalization_18 (Bat (None, 64)                256
chNormalization)
leaky_re_lu_18 (LeakyReLU)  (None, 64)                0
dense_12 (Dense)            (None, 4)                 260
=================================================================
Total params: 14,887,684
Trainable params: 172,868
Non-trainable params: 14,714,816
_________________________________________________________________
```

CNN Model 3

Training and Evaluation

- 20 epochs
- ‘Adam’ optimizer, learning rate = 0.001
- ‘categorical-crossentropy’ loss function, ‘accuracy’ metric
- ~50% accuracy on training, validation and unseen test data
- Generalized performance, not overfitted
- VGG model could achieve better accuracy since the accuracy plots has upward trajectory
- This will come at a cost of excessive computation time.



CNN Model 4

Architecture (Transfer Learning)

- **ResNet V2** convolutional and pooling layers imported
- Set transferred ResNet parameters untrainable
- 3 channel input (‘rgb’ colour mode)
- 3 extra dense layers, ‘relu’ activation (fully connected layers)
- BatchNormalization and Dropout layers for regularization
- ‘softmax’ activation for multi-class classification at output layer
- 2,138,948 trainable parameters
- 42,658,304 untrainable parameters

```
resnetmodel.summary()
```

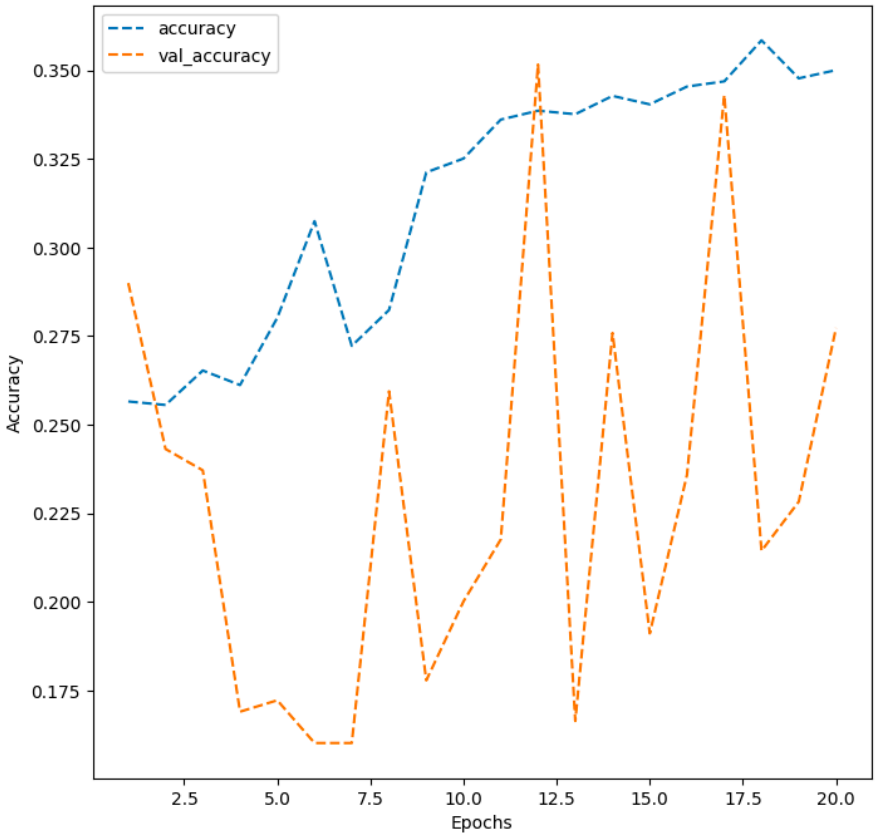
conv5_block2_3_conv (Conv2D)	(None, 2, 2, 2048)	1050624	['conv5_block2_2_relu[0][0]']
conv5_block2_3_bn (BatchNormalization)	(None, 2, 2, 2048)	8192	['conv5_block2_3_conv[0][0]']
conv5_block2_add (Add)	(None, 2, 2, 2048)	0	['conv5_block1_out[0][0]', 'conv5_block2_3_bn[0][0]']
conv5_block2_out (Activation)	(None, 2, 2, 2048)	0	['conv5_block2_add[0][0]']
conv5_block3_1_conv (Conv2D)	(None, 2, 2, 512)	1049088	['conv5_block2_out[0][0]']
conv5_block3_1_bn (BatchNormalization)	(None, 2, 2, 512)	2048	['conv5_block3_1_conv[0][0]']
conv5_block3_1_relu (Activation)	(None, 2, 2, 512)	0	['conv5_block3_1_bn[0][0]']
conv5_block3_2_conv (Conv2D)	(None, 2, 2, 512)	2359808	['conv5_block3_1_relu[0][0]']
conv5_block3_2_bn (BatchNormalization)	(None, 2, 2, 512)	2048	['conv5_block3_2_conv[0][0]']
conv5_block3_2_relu (Activation)	(None, 2, 2, 512)	0	['conv5_block3_2_bn[0][0]']
conv5_block3_3_conv (Conv2D)	(None, 2, 2, 2048)	1050624	['conv5_block3_2_relu[0][0]']
conv5_block3_3_bn (BatchNormalization)	(None, 2, 2, 2048)	8192	['conv5_block3_3_conv[0][0]']
conv5_block3_add (Add)	(None, 2, 2, 2048)	0	['conv5_block2_out[0][0]', 'conv5_block3_3_bn[0][0]']
conv5_block3_out (Activation)	(None, 2, 2, 2048)	0	['conv5_block3_add[0][0]']
flatten_4 (Flatten)	(None, 8192)	0	['conv5_block3_out[0][0]']
dense_13 (Dense)	(None, 256)	2097408	['flatten_4[0][0]']
dense_14 (Dense)	(None, 128)	32896	['dense_13[0][0]']
dropout_15 (Dropout)	(None, 128)	0	['dense_14[0][0]']
dense_15 (Dense)	(None, 64)	8256	['dropout_15[0][0]']
batch_normalization_19 (BatchNormalization)	(None, 64)	256	['dense_15[0][0]']
leaky_re_lu_19 (LeakyReLU)	(None, 64)	0	['batch_normalization_19[0][0]']
dense_16 (Dense)	(None, 4)	260	['leaky_re_lu_19[0][0]']

```
Total params: 44,797,252
Trainable params: 2,138,948
Non-trainable params: 42,658,304
```

CNN Model 4

Training and Evaluation

- 20 epochs
- ‘Adam’ optimizer, learning rate = 0.001
- ‘categorical-crossentropy’ loss function, ‘accuracy’ metric
- ~32% accuracy on training and unseen test data
- Divergent training and validation accuracies.
- Model appears erratic during training



CNN Model 6

Architecture

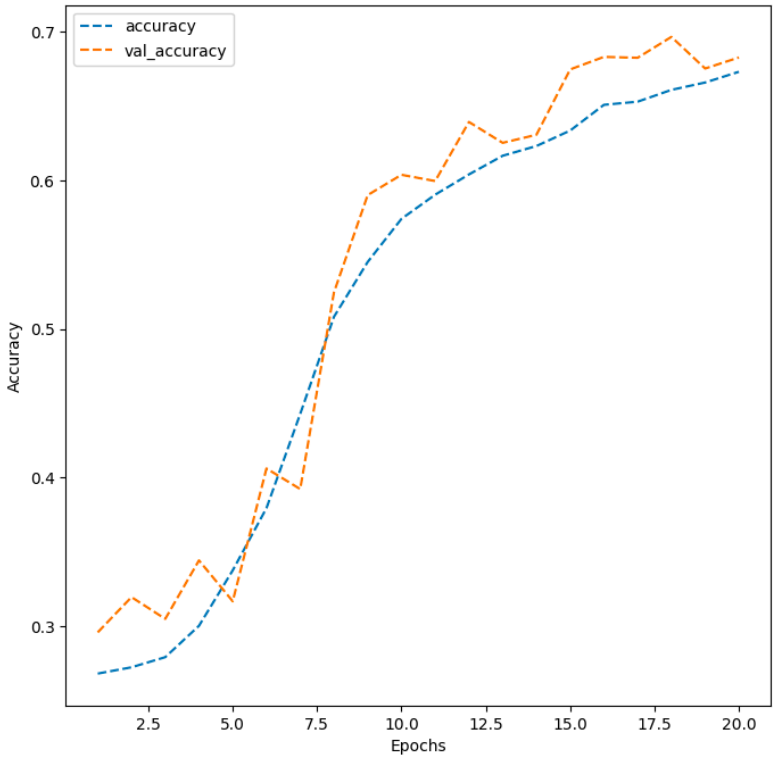
- 5 convolutional blocks
- Single channel input (‘grayscale’ colour mode)
- Maxpooling, ‘same’ padding, ‘relu’ activation
- BatchNormalization and Dropout layers for regularization
- 2 extra dense layers, ‘relu’ activation (fully connected layers)
- ‘softmax’ activation for multi-class classification at output layer

```
model3 = Sequential()

#1st CNN Block
model3.add(Conv2D(64, (2, 2), padding="same", input_shape=(48, 48, 1),activation = 'relu'))
model3.add(BatchNormalization())
model3.add(LeakyReLU(0.2))
model3.add(MaxPooling2D(2, 2))
model3.add(Dropout(0.2))



#2nd CNN Block
model3.add(Conv2D(128, (2, 2), padding="same",activation = 'relu'))
model3.add(BatchNormalization())
model3.add(LeakyReLU(0.2))
model3.add(MaxPooling2D(2, 2))
model3.add(Dropout(0.2))

#3rd CNN Block
model3.add(Conv2D(512, (2, 2), padding="same",activation = 'relu'))
model3.add(BatchNormalization())
model3.add(LeakyReLU(0.2))
model3.add(MaxPooling2D(2, 2))
model3.add(Dropout(0.2))

#4th CNN Block
model3.add(Conv2D(512, (2, 2), padding="same",activation = 'relu'))
model3.add(BatchNormalization())
model3.add(LeakyReLU(0.2))
model3.add(MaxPooling2D(2, 2))
model3.add(Dropout(0.2))

#5th CNN Block
model3.add(Conv2D(128, (2, 2), padding="same",activation = 'relu'))
model3.add(BatchNormalization())
model3.add(LeakyReLU(0.2))
model3.add(MaxPooling2D(2, 2))
model3.add(Dropout(0.2))

model3.add(Flatten())

# First fully connected layer
model3.add(Dense(256, activation = 'relu'))
model3.add(BatchNormalization())
model3.add(LeakyReLU(0.2))
model3.add(Dropout(0.2))

# Second fully connected layer
model3.add(Dense(512, activation = 'relu'))
model3.add(BatchNormalization())
model3.add(LeakyReLU(0.2))
model3.add(Dropout(0.2))

model3.add(Dense(no_of_classes, activation = 'softmax'))

model3.summary
```

CNN Model 6

Training and Evaluation

- 20 epochs
- ‘Adam’ optimizer, learning rate = 0.003
- ‘categorical-crossentropy’ loss function, ‘accuracy’ metric
- >64% accuracy on training, validation and unseen test data
- Generalized performance, not overfitted
- Model has the ability to achieve better accuracy since the accuracy plots appear to still be climbing.

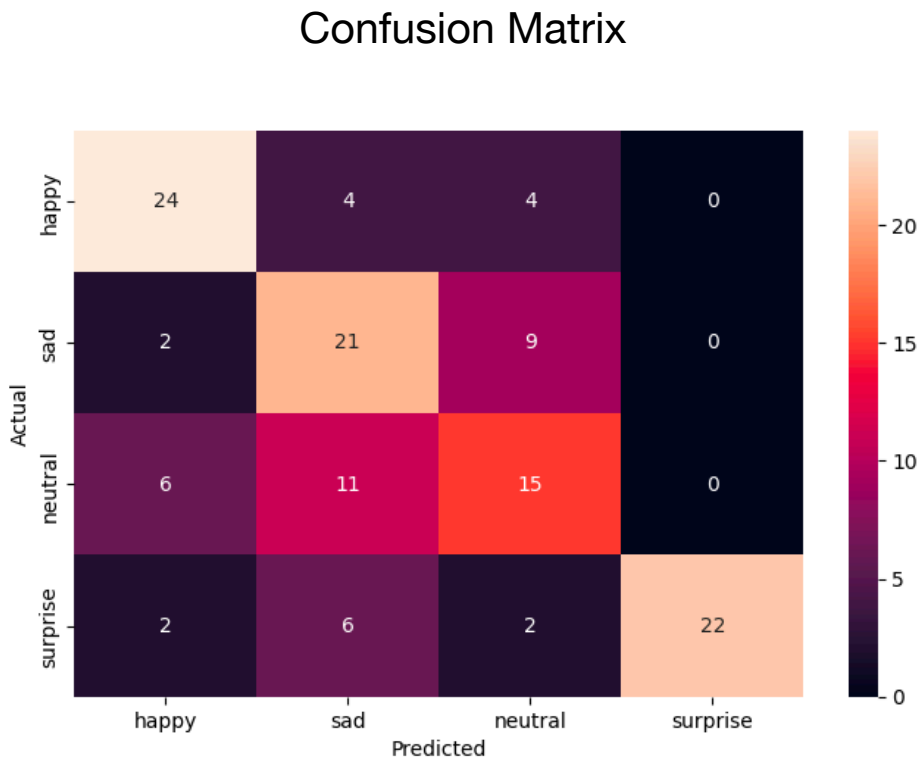


Classification Report

Chosen Model: Model 6

Unseen test data

- Accuracy: 64%
- Recall: 64%
- Precision: 68%
- f1-score: 65%
- 'neutral' has the most mis-classification error
- ‘surprise' has 100% precision



Insight and Recommendation

- Models 1, 2, 6 and the VGG model have the ability to achieve better accuracy since the accuracy plots appear to still be climbing as seen in the accuracy plots over epochs. This will come at a cost of excessive computation time. For this project, i stuck to 20 epochs for all NN.
- There is always a need to balance architecture complexity and number of training epochs to ensure model training is completed in reasonable time.
- Although Model2 has better accuracy than Model6, Model2 appears to be slightly overfitted as the training accuracy diverges from the validation accuracy.
- Model6 would be my preferred deep learning model for this project work