Deep Learning Approach for Facial Expression Recognition in Images with OCOsense Glasses-Worn Faces

Galena Dimeska.
Faculty of Electrical Engineering and Information Technologies,
University of Ss. Cyril and Methodius in Skopje, Macedonia

ABSTRACT

Deep learning is a very important type of field within the machine learning sphere that is extremely beneficial and instrumental for data science, especially where collecting, analyzing and interpreting large amounts of data is concerned. In this study, 10 different image classification models were defined and evaluated using deep learning methods, with the goal of finding the ones that would be best suited for facial expression recognition of a person wearing Emteq labs' OCOsense glasses. The initial data was extracted from the frames of the MOOD data collection videos of 20 subjects in the form of images. The data was then manually categorized into the desired classes (eyebrow_raise, frown, neutral, smile), and undersampling was performed on the majority class, neutral. The models were then defined, trained for the same number of epochs and evaluated in the same way. The results confirmed that simple models with few convolutional layers would perform worse in general, however a notable trend among the complex models where transfer learning was utilized showed that the results favored those models whose convolutional bases were fully trainable, as opposed to the ones where fine-tuning and freezing the lower layers was the technique used. The best model in terms of performance on the test set achieved an 85.44% accuracy.

KEYWORDS

facial expression, image, classification, model, accuracy, deep learning, transfer learning, convolutional base

1 Introduction

There is no disputing the significance of nonverbal behavior (NVB) and nonverbal communication (NVC) as they provide a wealth of information that can be learned about people through them. The face is regarded as the most significant NVB channel as it is the most sophisticated and complex nonverbal signal system in the body utilized by people to convey a wide range of their emotional and mental states [1]. A person's facial expressions oftentimes might provide insight into their thinking processes and psychological states. Reading someone's facial expressions allows one to understand how another person's mind is working through something, what their thoughts are focused on, and what their body is prepared to do in terms of action. Therefore, it is of no surprise that such a task would be a very popular topic for exploration in the deep learning and data science fields concerned with health informatics as well as life and medical sciences.

Even though there are many successful artificial intelligence algorithms and models that specialize in facial recognition and expression categorization, such as those used by mobile phone companies for unlocking their products with a camera scan of the owner's face, in general there have been difficulties in the past, as well as in the present, with facial detection when the examined person's face or head area isn't clear of any accessories such as glasses, hats, masks etc. This is due to the fact that it is almost always the case that such artificial intelligence models and algorithms are trained solely to classify or detect faces that are bare. For example, at the beginning of the Covid-19 pandemic, when it was necessary to protect oneself and others by wearing masks, the sudden influx of unexpected masked people presented great challenges to many widely used Facial Recognition (FR) systems, and sufficient adjustments had to be made to each in order to accommodate to the new situation.

Emteq labs has a wide variety of wearables, such as masks and glasses that are made to detect facial expressions and emotions. However, their great bulkiness and general presence on a person's head makes it difficult for an AI model to detect people's expressions, mood, or emotions correctly in pictures or videos when the person in question is wearing them, because they cover up a sizeable area of a person's eye, eyebrow and forehead area. This creates a problem as well as a need for such a model, as it would be greatly beneficial if such data could be analyzed and interpreted properly, to compare with the data and conclusions of Emteq's own paraphernalia, which would be helpful in improving the products. This study focuses specifically on creating and evaluating deep learning models on images of people wearing the Emteq OCOsense glasses in particular.

Many similar studies have been conducted with the theme of improving or creating facial recognition or facial expression detection systems on individuals wearing various head accessories. Zhen Li [2] investigated two new approaches to improve the performance of FR systems on masked facial images, both of which had promising results, and for the same purpose, Hariri Walid [3] proposed a reliable method based on discard masked region and deep learning-based features in order to address the problem of the masked face recognition process, showing high recognition performance. Liang et al. [4] addressed the challenge of performing face recognition on human faces that are wearing glasses and the experimental results for their proposed methods showed that the glasses detection rate was highly satisfactory for various face databases.

2 Data

For acquiring the necessary data, videos from the MOOD data collection by Emteq Labs. were used, specifically those from the task Forced Expressions, wherein various subjects who have depression wore Emteq's OCOsense glasses were filmed and were asked to watch 10 videos by which they were then instructed to make one of the following expressions – smile, frown, eyebrow raise. Every listed expression is repeated 3 times during each video.

From the entire MOOD data collection, a total of 20 subjects' videos were selected to be used for this study. Each video is in 30 frames per second and every other frame was extracted in order to garner the necessary data.

These images were then manually categorized into 4 different classes of mood/expression – eyebrow raise, frown, neutral and smile.

Through this process a huge number of images (approximately 195000) were extracted, with the dataset being heavily imbalanced due to the fact that "neutral" mood images overwhelmingly dominated it. This is due to the fact that in the filming process of the videos themselves, the subjects were given prompts with the goal of expressing a singular emotion, while being neutral in between each expression in every one of the videos. This has led to an overabundance of neutral mood images in the dataset, making it imbalanced. To help balance the dataset undersampling was performed by randomly selecting samples from the majority class, neutral, and deleting them from the dataset. The number of images labeled as neutral, which took up approximately 70% of the entire dataset, was sufficiently decreased, reducing the total dataset to 77900 images, the neutral class now making up 52% of the dataset.

These images were then split into training, validation and test datasets, with the purpose of being used for training multiple deep learning models, each category having a specific number of eyebrow raise, frown, neutral and smile images.

The number of images for each class in each subset can be seen in Table 1. Figure 1, on the other hand, depicts the percentages each of the classes takes up in the entire dataset.

	Images in Train set	Images in Validation set	Images in Test set
eyebrow_raise	4,500	700	1,000
frown	9,500	1,500	800
neutral	30,000	7,500	2900
smile	15,000	3,000	1,500
Total number	**59,000**	**12,700**	**6,200**

Table 1. Number of images per class in train, validation and test set.

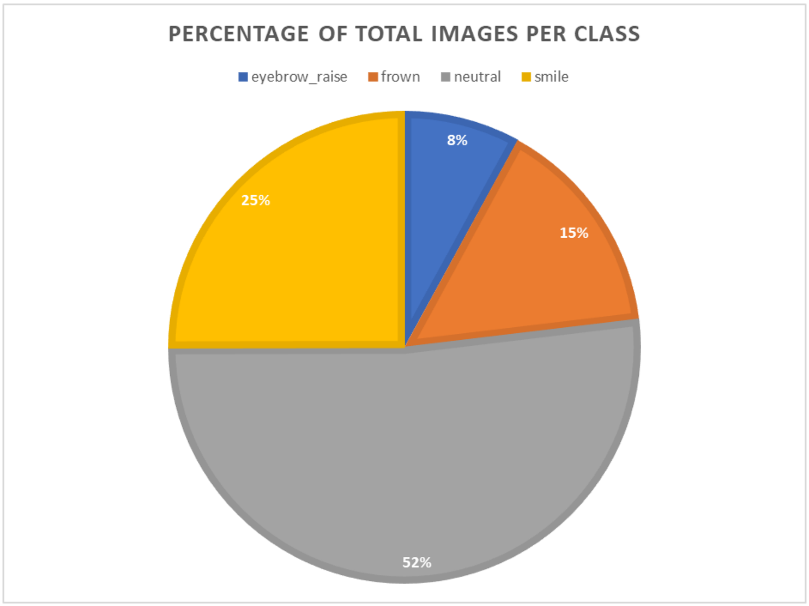

Figure 1. Depiction of the percentages of each of the classes within the full dataset

3 Approach and experiments

3.1 Data preprocessing

After setting the paths of the train, validation and test parent folders accordingly, image augmentation techniques were applied to the dataset in order to expand its size and incorporate more variation within it by mildly altering and transforming the existing images.

Specifically, the Keras ImageDataGenerator class was used because of the fact that it provides real-time data augmentation, by applying random transformations on each image as it is passed to the model and therefore ensuring that the model will receive new variations of the images at each epoch during its training phase.

The data augmentation functions applied to the training data are:

- Randomly flipping the inputs horizontally (horizontal_flip=True)
- Randomly rotating images up to 10 degrees (rotation_range=10)
- Randomly shifting input images left or right 10% of the image's width (width_shift_range=0.1)
- Randomly shifting input images up or down 10% of the image's height (height_shift_range=0.1)
- Randomly shearing the inputs (shear_range=0.01)
- Randomly zooming in on the images in the range [0.99; 1.01] (zoom_range=0.01)

The validation and test data were also augmented using the horizontal flip function.

In each of the train, validation and test data generators, the original RGB coefficients of the images, whose values range between 0 and 255, were rescaled by a factor of 1/255 for the models to be able to process the values, as they are originally too high. Furthermore, the target_size argument was set to resize the input images to a size of 244x244, and a batch size of 8 was chosen as well as the class mode "categorical" for 2D one-hot encoded labels.

3.2 Modeling

Several methods were used for training different models using deep learning techniques with python, such as training a model made from scratch, as well as (re)training and/or additional training of various reputable Keras image classification models that have achieved good results through transfer learning, namely fine-tuning, freezing layers as well as training the full convolutional base.

These Keras image classification models have been trained and tested for detecting and recognizing facial expressions and mood states of humans, performing quite successfully in previous experiments. Despite the fact that the training images to which the pretrained weights refer are of people not wearing any glasses or other paraphernalia covering the face, especially not ones of such a size as the OCOsense glasses, the dataset for this study is similar enough for transfer learning to seem like it would be a promising option, as it allows for the utilization of the already learned low-level features at the lower layers of these complex Keras models in the training process of the models in this experiment, with the hope of yielding decent results.

10 models in total were trained, one being a simple model made and trained completely from scratch, while transfer learning was used for training the remaining 9 models.

Several callback functions were defined for the models, identical for each of them:

- CSVLogger was used to stream epoch results into a CSV file.
- A ReduceLROnPlateau callback function was defined for reducing the learning rate when the chosen metric to monitor, once again validation loss, has stopped improving, with a patience of 6 epochs with no improvement before the learning rate would be reduced by a factor of 0.2.
- To save the "best" model, ModelCheckpoint was implemented, set to monitor the validation loss and only keep the version of the model which achieved the best results in performance, in this case the model in the state where the validation loss would be minimal in value. This "best" saved model was only additionally loaded and evaluated on the test data when the validation loss that was monitored was minimal in the very early few epochs of the training process, in order to compare the final accuracy and loss of that model to the ones of the model that would be fully trained for the full epochs.

Every model was compiled in the same manner. "Adam" was chosen as the gradient descent optimizer, as it was deemed an adequately efficient algorithm for learning the model weights while minimizing the total error on the training set for this problem/experiment. The learning rate was set to 0.0001, with a decay of 1e-6 (0.000001). CategoricalCrossentropy was used as a loss function due to the fact that there are more than 2 label classes, as well as accuracy as the metric, measuring what fraction of the images are correctly classified.

Additionally, all of the models were trained for 5 epochs each. At first it was attempted to train the first few models for 30 epochs, however upon reviewing the results and seeing that neither the accuracies not the losses of the train and validation sets improved at all after the first few epochs, despite the drastically long training period, it was decided to limit the number of epochs to 5 so as not to needlessly drain resources.

Regarding the 9 models to which transfer learning was applied, in all cases the tops of the pretrained models were not included and a new, and identical among every model, densely connected classifying block was defined and implemented on top in order to adapt the models so that they would be able to classify the images into the 4 classes established in this study (eyebrow_raise, frown, neutral and smile). The first layer in this classifier block is a Flatten layer, by which the output of the previous layer, i.e., the last layer of the convolutional base, is transformed into a one-dimensional array. The following layer is a Keras Dense layer, consisting of 512 neurons with a ReLU activation function, as well as a Dropout layer with a value of 0.5 applied below it, meaning that the probability of dropout would be 50%. The final layer is a 4-neuron Dense layer with a Softmax activation function that allows it to return an array of a total of 4 probability scores whose sum is 1, indicating the degree of likelihood of an image belonging to one of the 4 classes.

In all cases, the 9 models were initiated with ImageNet weights, and the input shape was set as (244,244,3) due to the fact that the images used were resized to 244x244 pixels and in RGB (3 color channels).

3.2.1 Simple model with 5 convolutional layers from scratch (Model 1)

The first model tested in these experiments, named “Model 1” is a relatively simple deep learning model which is comprised of 5 convolutional layers. All of the convolutional layers are 2D convolutional layers, the first of which has 32 filters, or kernels, with a size of 3x3, using a default stride value of 1x1, a ReLU activation function, input shape of (224,224,3), as explained before, and “SAME” padding, meaning that zero-padding will be added to the image accordingly in order for the layer’s outputs to have the same spatial dimensions as the inputs (this is possible because the stride is 1x1).

The second Conv2D layer is similar in that it has the same kernel size, padding and activation function, but with 64 kernels. A “BatchNormalization” layer is applied below it for normalizing the outputs from the convolutional layer before passing them on as the input of the next layer. A MaxPool2D layer follows, with a pool size of 2x2, for down-sampling the spatial dimensions of its input in order to reduce the number of parameters and computation in the network. It is followed by a Dropout layer with a 33% dropout rate.

The remaining convolutional layers are nearly identical, with the only difference being the number of filters – 128, 512 and 512, for each succeeding Conv2D layer respectively, each of which is followed by identical BatchNormalization, MaxPool2D and Dropout layers, as described previously.

The classification part consists of a few densely connected layers, beginning with a Flatten layer for transforming the output of the previous layer into a 1D array, followed by two Dense layers with 256 and 512 neurons respectively, and ReLU activation function, each of which are followed by a BatchNormalization layer, and a Dropout layer with a rate of 33%. The last layer is a 4-neuron Softmax Dense layer for finally classifying the input images into one of the 4 classes.

The compiled model has a total of 29,006,404 parameters, of which 29,002,436 are trainable and 3,968 are non-trainable.

After being trained for the 5 standard epochs, Model 1 was evaluated on each of the train, validation and test sets and achieved the following results:

- Final Train Accuracy = 50.30%
- Final Validation Accuracy = 40.20%
- Final Test Accuracy = 46.79%

The final test accuracy is considerably low, with the model successfully predicting the correct class of the images less than half of the time.

Figure 2 below depicts a more detailed look at Model 1’s performance during the training session, showing that the validation loss was minimal, as well as the validation accuracy maximal, after the 4th epoch (note that on the graphs the epochs are enumerated from 0-4, as opposed to their actual numbers, 1-5).

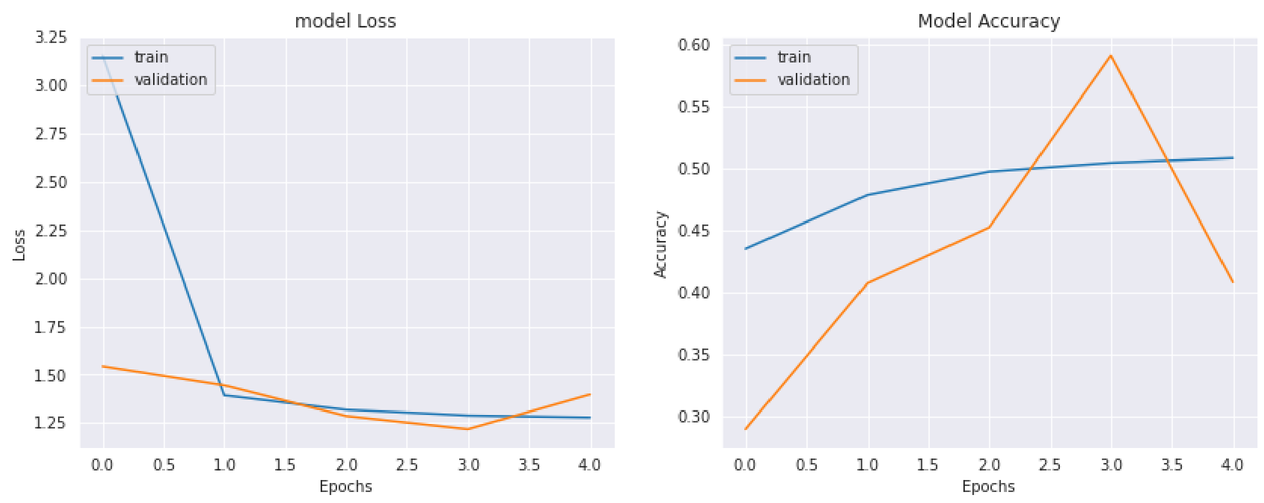

Figure 2. Depiction of Model 1's loss and accuracy throughout the training process.

3.2.2 MobileNetV2 (Model 2)

MobileNetV2 was used as a convolutional base for the second model, named "Model 2". It is a successful image classification model [5] developed by researchers at Google, similar to MobileNet, but with a drastically lower parameter count and using inverted residual blocks with bottlenecking features. MobileNetV2 has achieved a top-5 accuracy of 90.1% and has a total of 3,538,984 parameters.

For the construction of Model 2, the MobileNetV2 base was fine-tuned, freezing the blocks of the convolutional base up until the one named "block_10_expand", and unfreezing the others, and then adding the previously described new classifier block on top.

The compiled model2 has a total of 4,053,548 parameters, of which 3,780,076 are trainable and 273,472 non-trainable.

After the 5-epoch training period, model2 was evaluated on each of the train, validation and test sets and achieved the following results:

- Final Train Accuracy = 94.66%
- Final Validation Accuracy = 69.56%
- Final Test Accuracy = 71.95%

The final test accuracy is relatively decent, showing a considerable improvement from Model 1.

Figure 3 below presents Model 2's performance regarding its train and validation accuracy and loss during the training session. The validation loss was minimal (1.18908) after the final, 5th, epoch, and the validation accuracy was at its maximum (0.6985) also at that point.

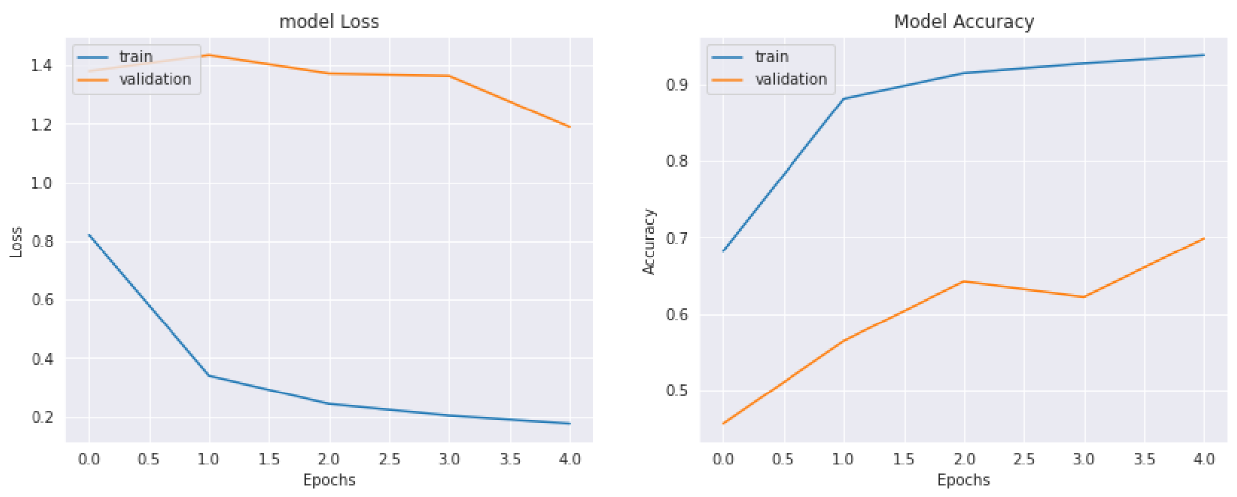

Figure 3. Depiction of Model 2's loss and accuracy throughout the training process.

3.2.3 VGG16Imagenet (Model 3)

For the third model, "Model 3", VGG16 was used as a convolutional base. It is a successful and very popular image classification model [5] developed in 2014 by Karen Simonyan and Andrew Zisserman of the Visual Geometry Group Lab of Oxford University. It has a depth of 16, and has achieved a top-5 accuracy of 90.1%, with a total of 14,714,688 parameters.

While fine-tuning the VGG16 base, the blocks up until the one named "block3_conv1" were frozen, while allowing the others to be trainable. The previously described new classifier block for this study's problem was added on top in order to finalize the architecture of Model 3. As stated before, ImageNet weights were used.

Model 3 has a total of 27,562,308 parameters, of which 27,302,148 are trainable and 260,160 non-trainable.

After training model3 for 5 epochs, it was evaluated on each of the train, validation and test sets and achieved the following results:

- Final Train Accuracy = 87.72%
- Final Validation Accuracy = 67.69%
- Final Test Accuracy = 74.1%

The final test accuracy is also relatively decent.

Model 3's performance regarding its train and validation accuracies and losses during its training period may be seen in Figure 4. The validation loss was at its minimum (0.99719) after the 5th epoch, and the validation accuracy was also the highest (0.6756) at that point.

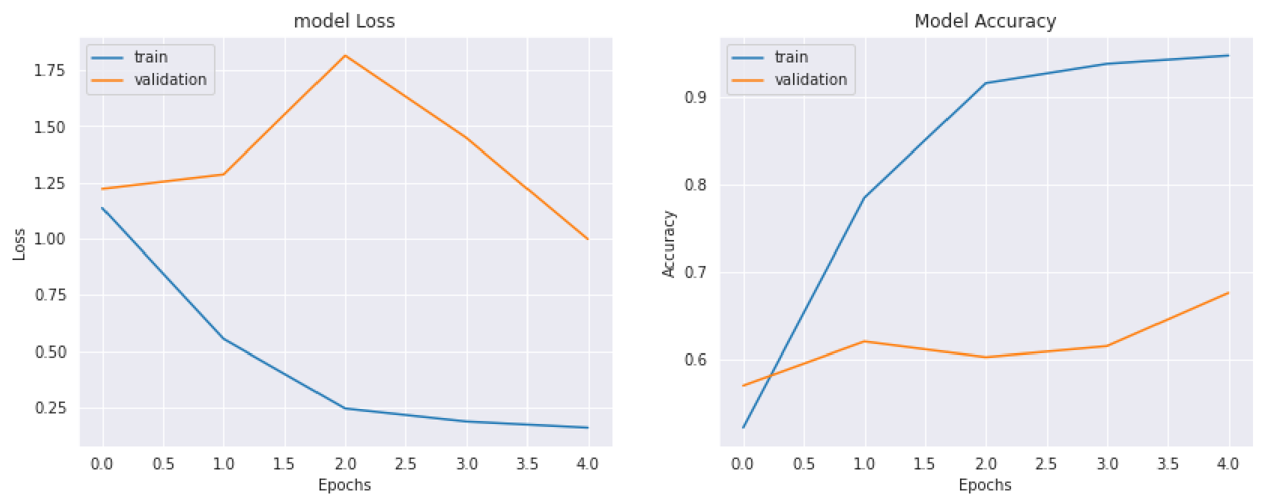

Figure 4. Depiction of Model 3's loss and accuracy throughout the training process.

3.2.4 InceptionResNetV2 (Model 4 and Model 4_2)

The following 2 models use InceptionResNetV2 as a base. It is a very efficient and successful Keras image classification model [5] with a total of 55,873,736 parameters and has achieved a top-5 accuracy of 95.3%.

3.2.4.1 Model 4

For the fourth model, "Model 4", the InceptionResNetV2 base was fine-tuned in such a way where the blocks up until the one named "conv2d_53" were frozen, setting the other remaining layers as trainable. To finish off designing Model 4, the new classifier block described earlier on was added on top.

The compiled final Model 4 has 55,115,708 trainable, and 1,272,592 non-trainable parameters, bringing the total number of parameters to 56,388,300.

Model 4 was then trained for 5 epochs and after completing its training, it was evaluated on each of the train, validation and test sets, achieving the following results:

- Final Train Accuracy = 97.18%
- Final Validation Accuracy = 65.13%
- Final Test Accuracy = 78.26%

The final test accuracy is reasonably adequate.

The performance of Model 4 during its training period, in terms of the train and validation accuracies and losses it achieved, is shown in Figure 5. The validation loss was minimal (1.15945) after the 5th epoch, while, on the other hand, the validation accuracy was actually the highest (0.6575) after the 3rd epoch.

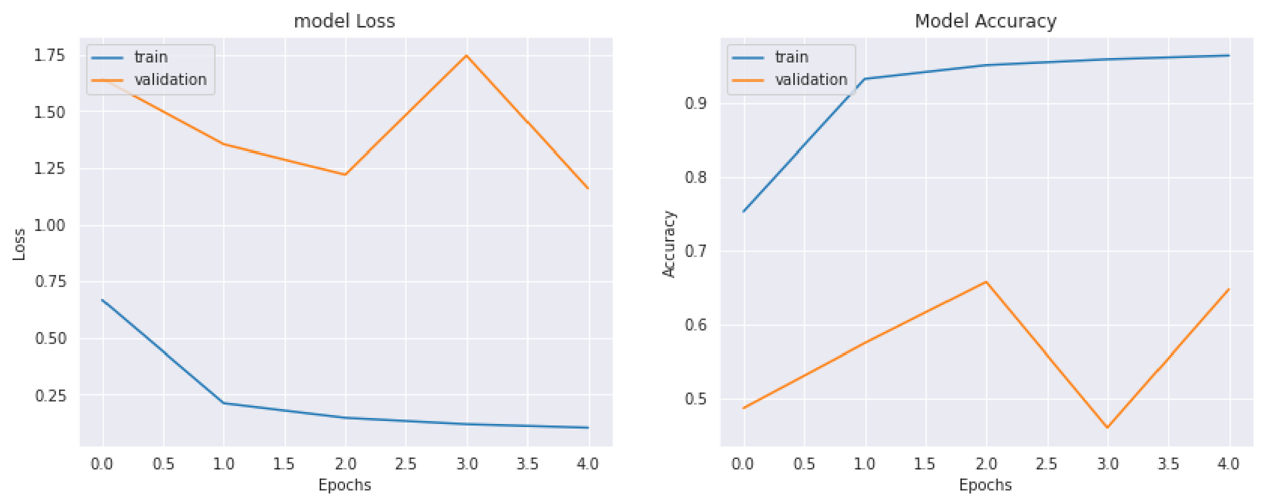

Figure 5. Depiction of Model 4's loss and accuracy throughout the training process.

3.2.4.2 Model 4.2

An additional model, "Model 4.2", was created using InceptionResNetV2 as its base. Model 4.2 uses the same classifier block on top, but unlike Model 4, the entire convolutional base was set as trainable with the aim of comparing the results of 2 models that have the same base, but different trainability in layers.

Model 4.2 has 56,388,300 parameters in total, the same number as model4, but this time 56,327,756 of those are trainable and 60,544 non-trainable.

After being trained for 5 epochs and evaluated on each of the train, validation and test sets, Model 4.2 achieved the following results:

- Final Train Accuracy = 97.06%
- Final Validation Accuracy = 40.82%
- Final Test Accuracy = 80.13%

The final test accuracy is fairly good, and a slight improvement from Model 4's. Model 4.2's performance while training is depicted in Figure 6.

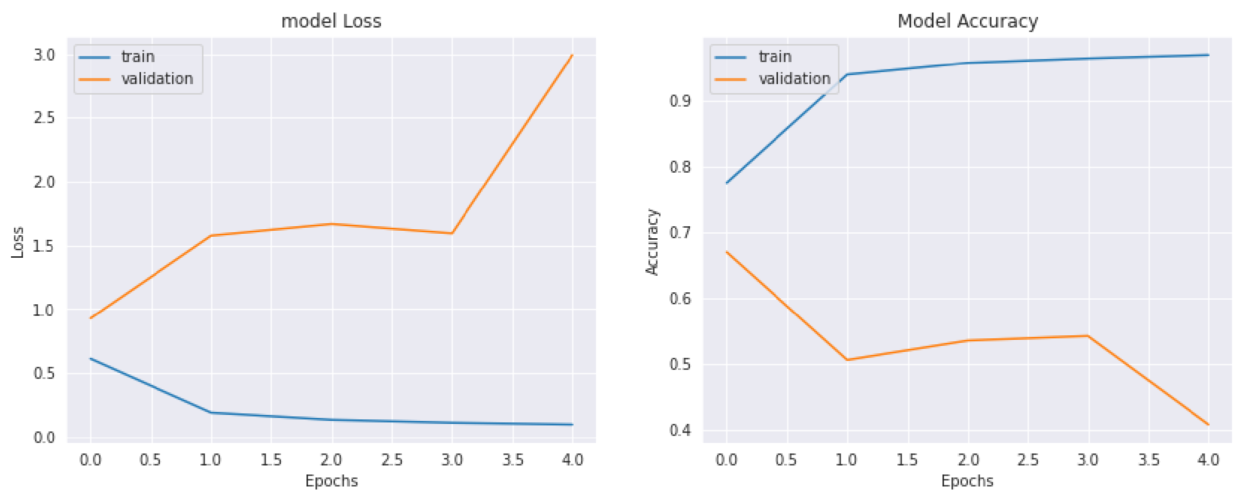

Figure 6. Depiction of Model 4.2's loss and accuracy throughout the training process.

The validation loss was minimal (0.92769) at the very beginning and kept increasing over time continuously, while the validation accuracy was the highest (0.6696) at the same point.

Due to the fact that the validation loss, was the lowest at the very start of the training session, the ModelCheckpoint callback function saved that version of the model as the "best" model, and this version was additionally loaded and evaluated on the validation and test data, with the following results:

- Final Validation Accuracy = 67.13%
- Final Test Accuracy = 83.69%

Compared to the fully trained model4, the validation accuracy increased quite a bit, over 26%, and there was also a slight betterment of the test accuracy as well.

3.2.5 MobileNetV3Large (Model 5)

MobileNetV3 in itself is defined as two models, namely, MobileNetV3-Small (for low resource uses), and MobileNetV3- Large (for high resource uses), which is the one used in these experiments. Compared to MobileNetV2, MobileNetV3-Large is 3.2% more accurate on ImageNet classification while additionally reducing latency by around 20% [6]. It has a total of 5,507,432 parameters.

In the design of the fifth model, named "Model 5", the MobileNetV3Large convolutional base was used and fine-tuned. The blocks of the convolutional base up until the one named "expanded_conv_6/expand" were frozen, the rest remaining trainable. The new classifier block was added on top of the base.

Model 5 has 6,021,996 parameters in total, 5,936,476 of which are trainable and 85,520 are non-trainable.

After the 5-epoch training period, model5 was evaluated on each of the train, validation and test sets and the following results were achieved:

- Final Train Accuracy = 88.91%
- Final Validation Accuracy = 53.76%
- Final Test Accuracy = 51.13%

The final test accuracy is very mediocre, the model successfully predicting the correct class of the images a little bit more than half of the time.

Figure 7 below presents Model 5's performance during the training session.

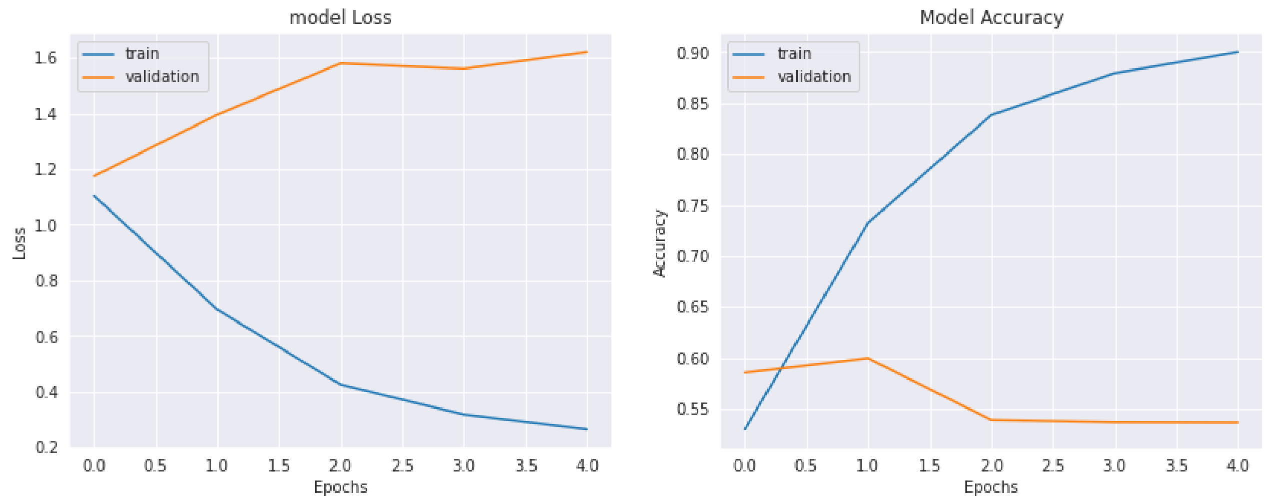

Figure 7. Depiction of Model 5's loss and accuracy throughout the training process.

It can be observed that in this case, the validation loss was minimal (1.17457) after only the first epoch and kept increasing over time continuously, while the validation accuracy was the highest (0.6985) at the 2nd epoch.

Due to the fact that the validation loss, monitored by the "ModelCheckpoint" callback function, was the lowest at the very beginning of the training session, the saved "best" model, according to this performance of Model 5, was additionally loaded and evaluated on the validation and test data, with the following results:

- Final Validation Accuracy = 58.28%
- Final Test Accuracy = 55.61%

There is a slight increase in both of the accuracies compared to the fully trained Model 5.

3.2.6 EfficientNet (Model 6 and Model 7)

EfficientNets are a family of models [7] developed in 2019 by Mingxing Tan and Quoc V. Le. EfficientNets are suitable for transfer learning and have achieved state-of-the-art accuracy on Flowers (98.8%), CIFAR-100 (91.7%), and other transfer learning datasets. EfficientNet-B0, in particular, was used for the following 2 models, with 4,049,564 parameters, and it has achieved top-1 accuracy of 77.1% and top-5 accuracy of 93.3% on ImageNet [5].

3.2.6.1 Model 6

The seventh model, “Model 6”, was designed with an EfiicientNetB0 base. The fine-tuning technique was used and the blocks up until the one named "block4a_expand_conv" were frozen, while the other remaining layers were set as trainable. The standard classifier block for this study was added on top.

The compiled final model6 has 36,057,022 trainable, and 107,746 non-trainable parameters, bringing the complete number of parameters to 36,164,768.

Model 6 was trained for the standard 5 epochs and after completing the training, it was evaluated on each of the train, validation and test sets, with these results:

- Final Train Accuracy = 50.85%
- Final Validation Accuracy = 59.06%
- Final Test Accuracy = 46.77%

The final test accuracy is considerably low, lower than even the simple Model 1 with its 5 convolutional layers, in fact. The model successfully predicts the correct class of the images less than half of the time.

The performance of Model 6 in terms of the train and validation accuracies and losses it achieved while being trained, is presented in Figure 8. The validation loss was minimal (1.07844) at the 2nd epoch, while the validation accuracy was actually the highest (0.5906) after the 1st, 2nd and 4th epoch, being of identical value in each case.

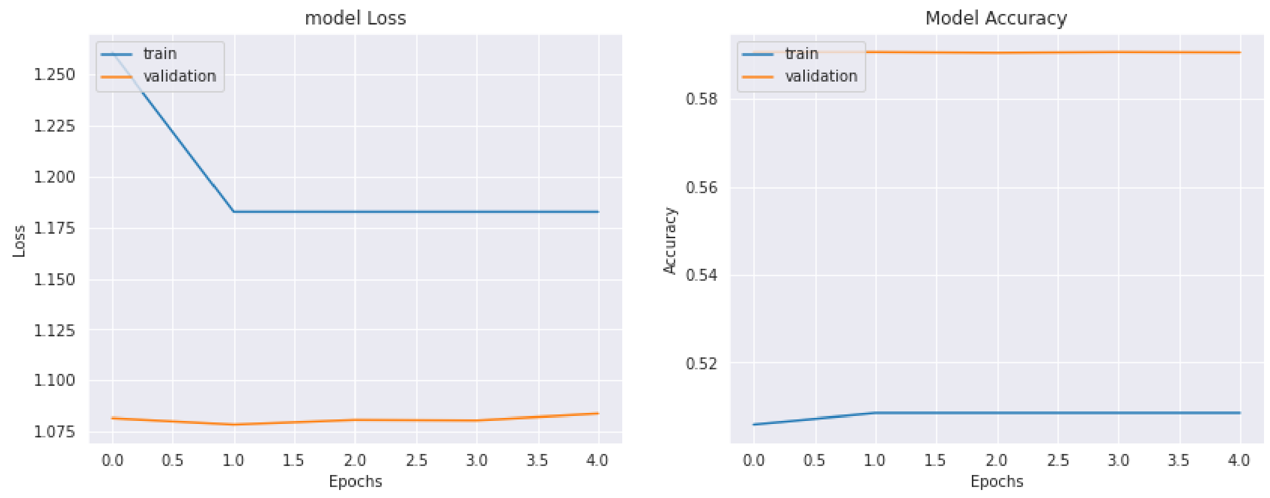

Figure 8. Depiction of Model 6’s loss and accuracy throughout the training process.

3.2.6.1 Model 7

Using the same EfficientNetB0 model as a base, “Model 7” was designed, and in this iteration the entire convolutional base with all its layers was set to be fully trainable.

Model 7 has 36,164,768 total parameters, the same as Model 6, however, Model 7 has 36,122,752 trainable and 42,016 non-trainable parameters.

After the 5-epoch training period, Model 7 was evaluated on the train, validation and test sets, achieving the following results:

- Final Train Accuracy = 96.32%
- Final Validation Accuracy = 55.24%
- Final Test Accuracy = 84.47%

The final test accuracy is fairly good, and a slight improvement from model4's.

Figure 9 below showcases Model 7's performance during its training session.

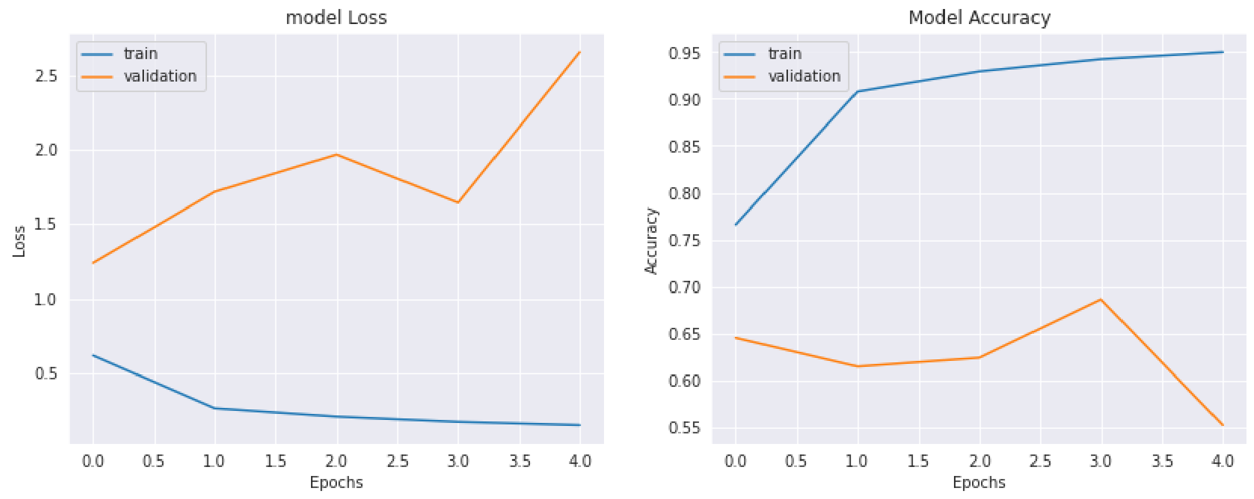

Figure 9. Depiction of Model 7's loss and accuracy throughout the training process.

The validation loss was minimal (1.23847) at the 1st epoch and only continued increasing. The validation accuracy was the highest (0.6864) at the 4th epoch.

Once again, the "best" model version saved by the ModelCheckpoint callback function was loaded and evaluated on the validation and test data, with these results:

- Final Validation Accuracy = 64.32%
- Final Test Accuracy = 77.95%

While the validation accuracy improved from the fully trained Model 7, the test accuracy score actually worsened.

3.2.7 InceptionV3 (Model 8)

InceptionV3 was used as a base for the penultimate model in this study, “Model 8”. It is yet another image classification model, particularly the third edition of the Inception Convolutional Neural Network by Google, having 21,802,784 parameters in the subnetwork used for this study.

Due to the fact that in the previous tastings, the models in which the base was set to be fully trainable achieved better results, it was decided to do the same with model8.

The fully designed and compiled Model 8 has 48,019,748 parameters, 47,985,316 trainable and 34,432 non-trainable.

Model 8 was evaluated on the train, validation and test sets after it completed its 5-epoch training with the following results:

- Final Train Accuracy = 95.82%
- Final Validation Accuracy = 68.34%
- Final Test Accuracy = 85.44%

The final test accuracy is quite good, achieving the highest test accuracy out of all the models.

Figure 10 below presents Model 8’s performance during the training period of the model.

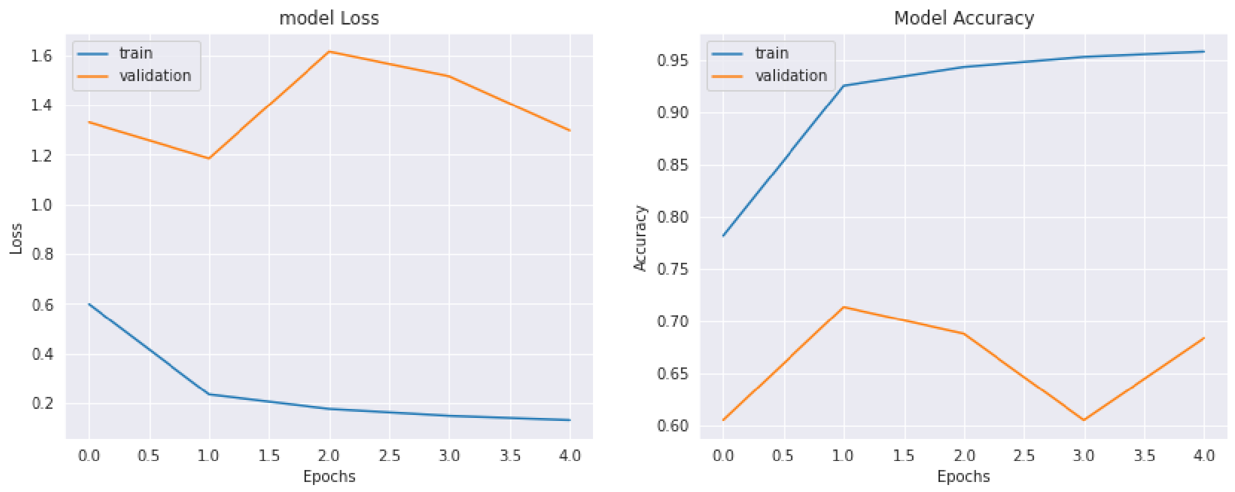

Figure 10. Depiction of Model 8’s loss and accuracy throughout the training process.

The validation loss was minimal (1.18521) at the 2nd epoch and then increased. The validation accuracy was the highest (0.7132) at the same epoch.

Yet again, the “best” model version saved by the ModelCheckpoint callback function was loaded and evaluated on the validation and test data, with the following results:

- Final Validation Accuracy = 71.49%
- Final Test Accuracy = 84.18%

Although the validation accuracy improved, the test accuracy decreased slightly.

3.2.8 Xception (Model 9)

Xception is another Keras image classification model, originally developed by François Chollet in 2017 [8]. It has achieved top-1 accuracy of 79% and top-5 accuracy of 94.5%, and has 20,861,480 parameters. An Xception base was used for the final model evaluated in this study, "Model 9".

Once again, the base was set to be fully trainable. As such, after adding the standard classifying block, Model 9 has an astounding 72,244,268 parameters, 72,189,740 trainable and 54,528 non-trainable.

Following the completion of its training, model9 was evaluated on the train, validation and test sets, achieving these results:

- Final Train Accuracy = 96.61%
- Final Validation Accuracy = 70.46%
- Final Test Accuracy = 83.87%

The final test accuracy is relatively good.

Model 9's performance regarding its train and validation accuracy and loss during the training session can be seen in more detail in Figure 11. The validation loss was the lowest (1.04681) at the 2nd epoch, meanwhile the validation accuracy was maximal (0.7006) at the last epoch.

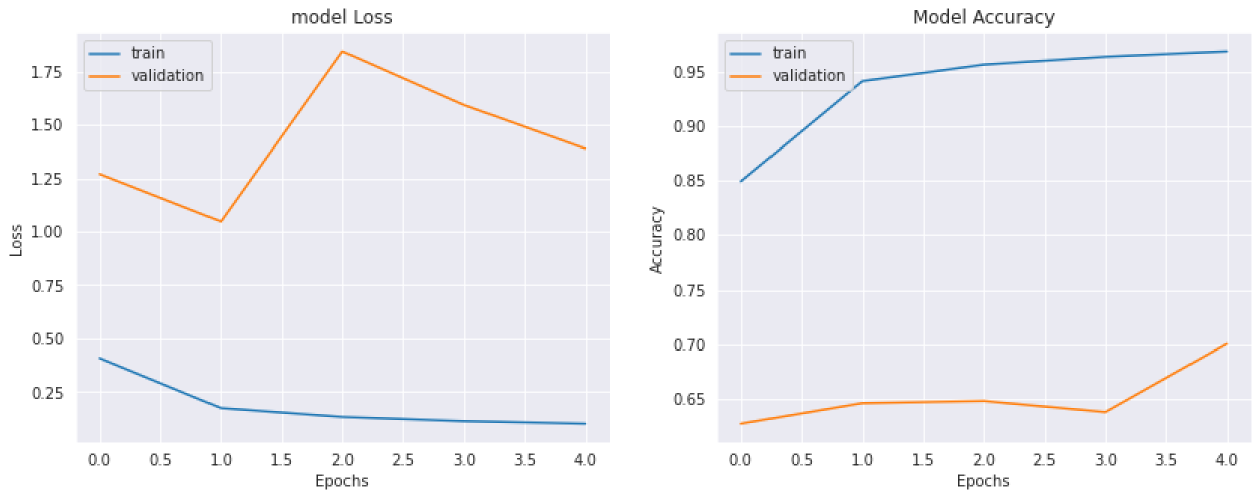

Figure 11. Depiction of Model 9's loss and accuracy throughout the training process.

3.3 Overall results and comparisons

The final accuracies for the train, validation and test sets computed for each model in this study are summarized in Table 2 below.

Model name	Conv. base	Final Train Accuracy (%)	Final Validation Accuracy (%)	Final Test Accuracy (%)
Model 1	/	**50.3**	**40.2**	**46.79**
Model 2	MobileNetV2	**94.66**	**69.56**	**71.95**
Model 3	VGG16Imagenet	**87.72**	**67.69**	**74.1**
Model 4	InceptionResNetV2	**97.18**	**65.13**	**78.26**
Model 4_2	InceptionResNetV2	**97.06**	**40.82**	**80.13**
Model 5	MobileNetV3Large	**88.91**	**53.76**	**51.13**
Model 6	EfficientNet	**50.85**	**59.06**	**46.77**
Model 7	EfficientNet	**96.32**	**55.24**	**84.47**
Model 8	InceptionV3	**95.82**	**68.34**	**85.44**
Model 9	Xception	**96.61**	**70.46**	**83.87**

Table 2. Train, validation and test accuracies of each of the 10 models.

The worst-performing model was surprisingly, not the simple model with 5 convolutional layers, but Model 6, which tried to leverage transfer learning by fine-tuning an EfficientNet base and freezing some of the bottom blocks, achieving a measly test accuracy of 46.79%.

The model that performed the best on the test dataset is Model 8, which used a fully trainable InceptionV3 architecture, achieving an accuracy score of 85.44%.

The confusion matrix of the validation set of Model 8 is depicted below, in Table 3, while the one on the test set in Table 4. The labels in the first column represent the true labels, while those in the first row refer to the predicted labels. The values that are bolded represent the number of images that were correctly classified, i.e., the amount of time the predicted label matched the true label.

Confusion matrix	eyebrow_raise	frown	neutral	smile
eyebrow_raise	**38**	135	416	111
frown	70	**234**	941	255
neutral	270	1408	**4667**	1155
smile	114	535	1912	**439**

Table 3. Confusion matrix of validation set of Model 8.

Confusion matrix	eyebrow_raise	frown	neutral	smile
eyebrow_raise	**92**	136	576	196
frown	76	**74**	447	173
neutral	304	295	**1693**	608
smile	163	159	889	**289**

Table 4. Confusion matrix of test set of Model 8.

It is clear by looking at both of the confusion matrixes that the model tended to misclassify the images from the eyebrow_raise, frown and smile classes the most into the neutral class. This further exaggerates the fact that the dataset is imbalanced, and despite the undersampling done, there are still a lot more neutral images in the dataset, leading to results such as these.

Furthermore, it is important to note that between all of the models that used an existing Keras image classification model as a base, the ones where the entire convolutional base, with all of its layers, was defined to be fully trainable, greatly outperformed those models which utilized fine-tuning and froze some of the lower layers of the base they used.

4 Conclusion

This study explored and investigated the potential of the utilization of deep learning for facial expression recognition of individuals wearing Emteq's OCOsense glasses on several image classifying models. A vast amount of data was extracted in image form from videos of 20 participants that watched a total of 10 videos by which they were asked make a specific facial expression. The models performed either mediocrely or sufficiently good (see Table 2), however there were no outstanding performers. The best model was model8, with a fully trainable InceptionV3 base architecture and a custom dense classifying block, achieving an accuracy of 85.44% on the test data. The confusion matrixes provided insight into which of the classes the models favored to classify an image into, that being "neutral", which is most likely due to the fact that the subjects' expressions may have been very subtle at times and barely distinguishable from their neutral mood, as well as the fact that the dataset was imbalanced in favor of the neutral class, in spite of the undersampling done to reduce this. In the future, even more complex models may be explored in order to find a satisfactory solution, as it would greatly assist in the analysis and betterment of Emteq's product by providing a way to compare the results of the deep learning models to the OCOsense glasses' own data and conclusions, and seeing what improvements may be made, as nothing can be 100% accurate.

REFERENCES

[1] David Matsumoto, Benefits of Reading Facial Expressions of Emotion, 6 April, 2021

[2] Zhen Li, “Facial Recognition for People Wearing Masks,” Department of Computer Science, Stanford University

[3] Walid, Hariri, “Efficient Masked Face Recognition Method during the COVID-19 Pandemic,” Department of Computer Science, Badji Mokhtar Annaba University, 2020

[4] A. Liang, C. S. N. Pathirage, C. Wang, W. Liu, L. Li and J. Duan, "Face Recognition Despite Wearing Glasses," 2015 International Conference on Digital Image Computing: Techniques and Applications (DICTA), Adelaide, SA, Australia, 2015

[5] https://keras.io/api/applications/ 12 February 2023

[6] Howard, Andrew & Pang, Ruoming & Adam, Hartwig & Le, Quoc & Sandler, Mark & Chen, Bo & Wang, Weijun & Chen, Liang-Chieh & Tan, Mingxing & Chu, Grace & Vasudevan, Vijay & Zhu, Yukun, “Searching for MobileNetV3,” 2019

[7] Mingxing Tan, Quoc V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” 11 Sep 2020

[8] François Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” 4 Apr 2017