FOUNDATIONS OF DEEP LEARNING

FLOWER CLASSIFICATION

Yuliia Tsymbal 894213
Yasmin Bouhdada 837389
Paola Cavana 859341

INTRODUCTION

The chosen dataset contains 3670 images of flowers, in particular of the following **5 categories**:

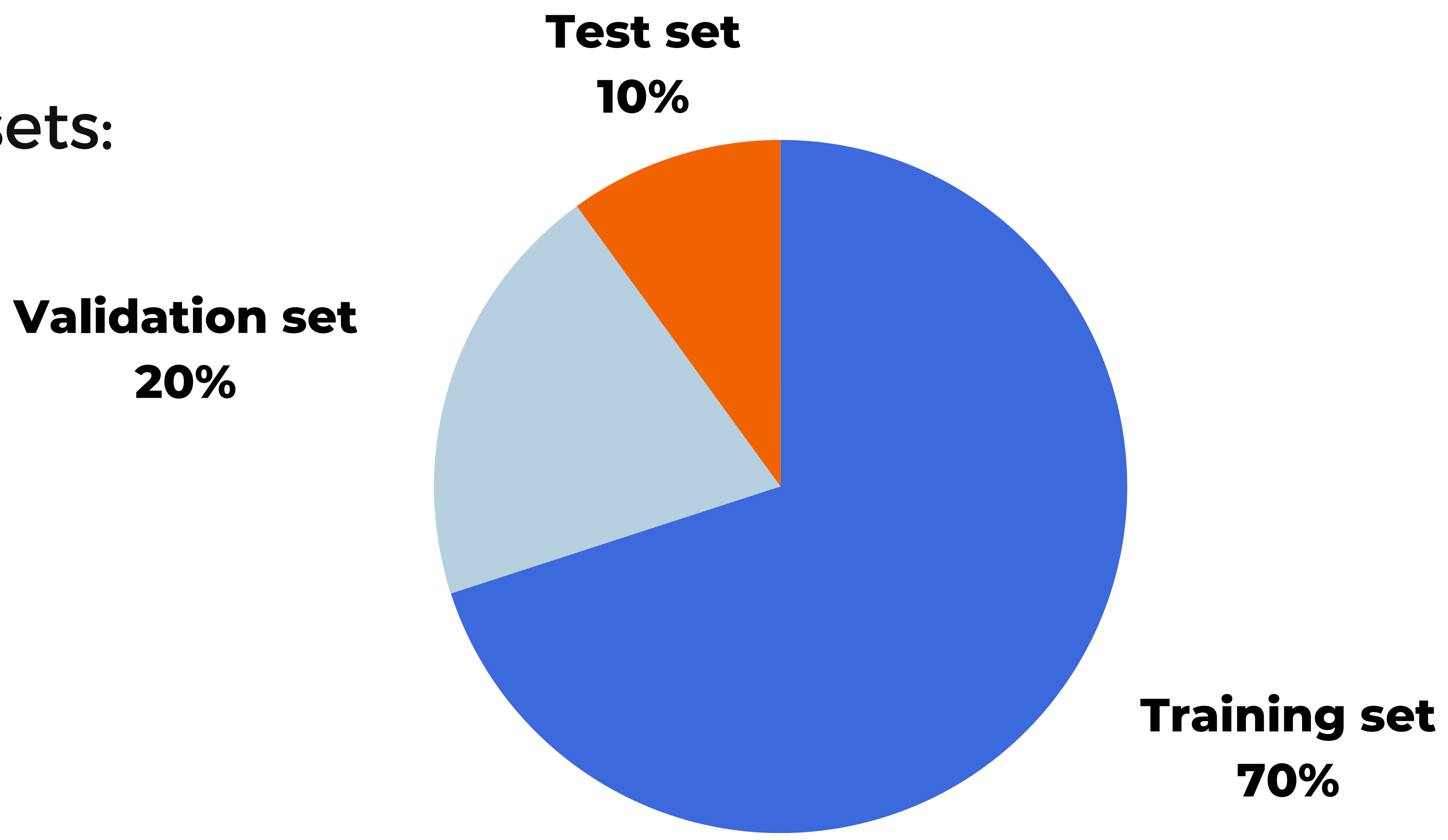
- Daisy (633 images)
- Dandelion (898 images)
- Roses (641 images)
- Sunflowers (699 images)
- Tulips (799 images)

Our goal is to **classify** the images contained in this dataset, in their respective categories.

DATA PREPARATION

At this stage we divided our dataset into three sets:

- Training set (70%)
- Validation set (20%)
- Test set (10%)



We also set the following parameters:

- Batch size: **32**
- Image size: **(224, 224)**

As final steps of this part, we have applied the following techniques:

- **Data augmentation**
- **Data segmentation**

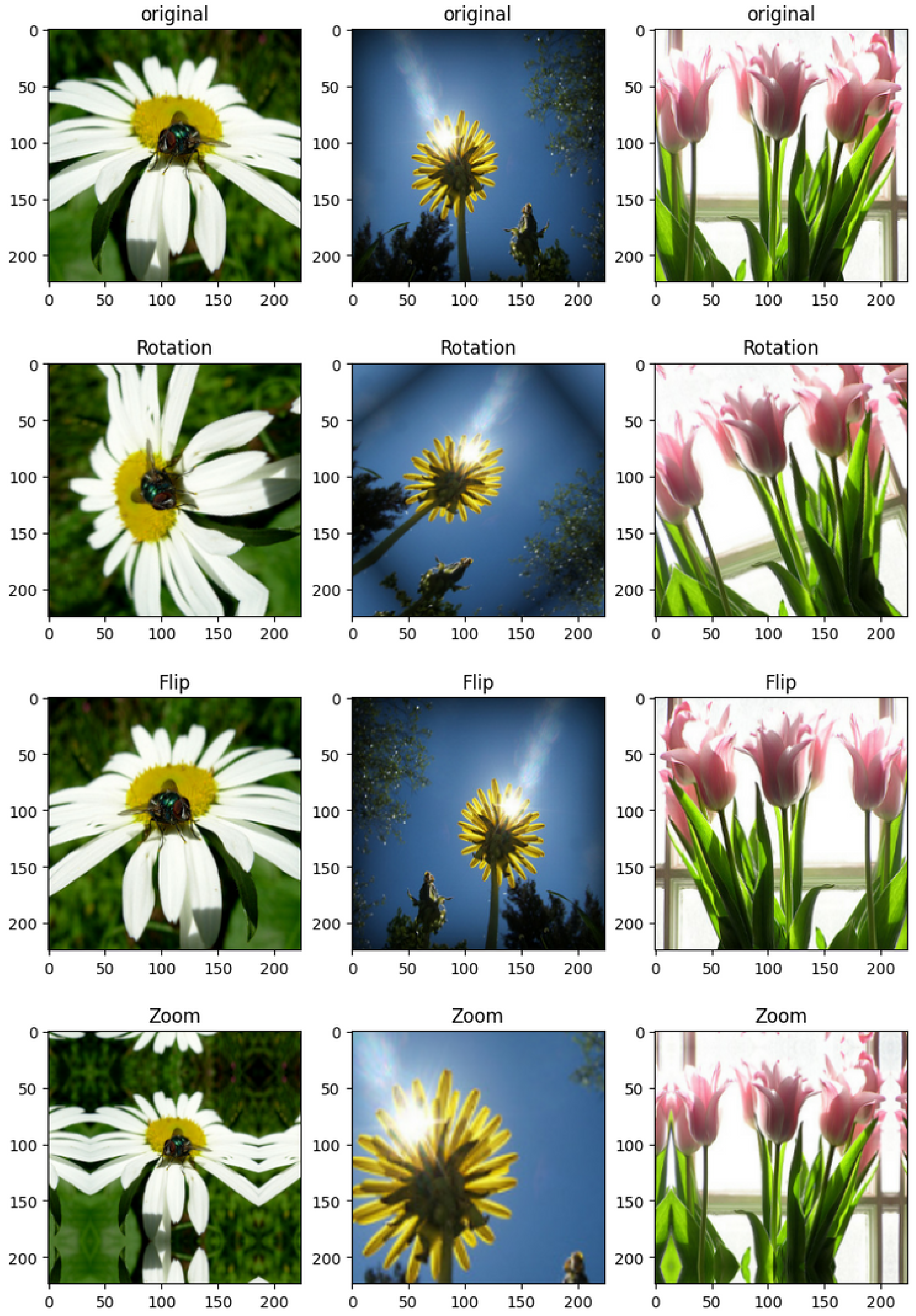
DATA AUGMENTATION

As we know the Data Augmentation is a technique to artificially increase the training set by creating modified copies of a data set using existing data.

We have applied this technique as it is useful to avoid the overfitting of models.

In particular we have modified the images in these aspects:

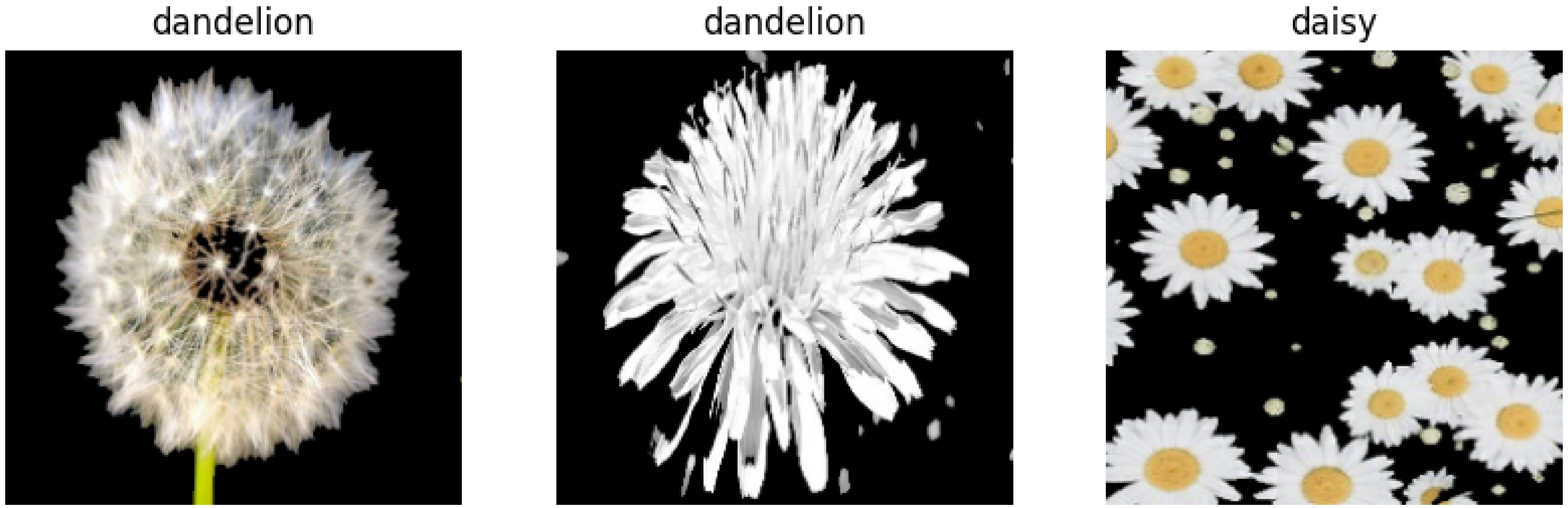
- Rotation
- Flip
- Zoom



DATA SEGMENTATION

In an image of this dataset it is possible that there are too many elements that could confuse a model. For this, we have used a segmentation technique to segment the dataset.

We have obscured everything in the image that is not part of the flower.



MODEL SELECTION

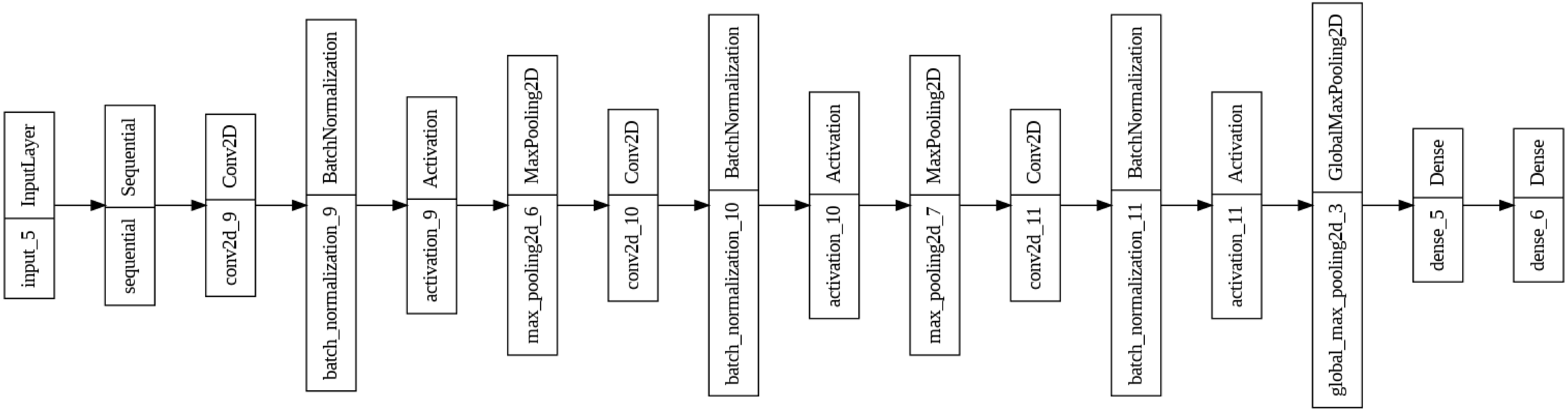
To achieve our goal, we have developed three different solutions.

- Firstly, we realized a **Convolutional Neural Networks** (CNN), commonly used for image processing tasks.
- Secondly, we decided to use the **transfer learning** approach with the pre-trained model **ResNet-50**.
- And as a third option, we considered the **fine-tuning** approach with the pre-trained model **VGG-16**.

MODELS FROM SCRATCH

Model 1 with 3 convolution layers with batch normalization and activation function 'Relu', 2 dense layers and kernel regularizer.

Total params: 102,725
Trainable params: 102,277
Non-trainable params: 448



EVALUATION

Model 1 + data augmentation on original dataset

- **Training**:

Loss: 0.6754

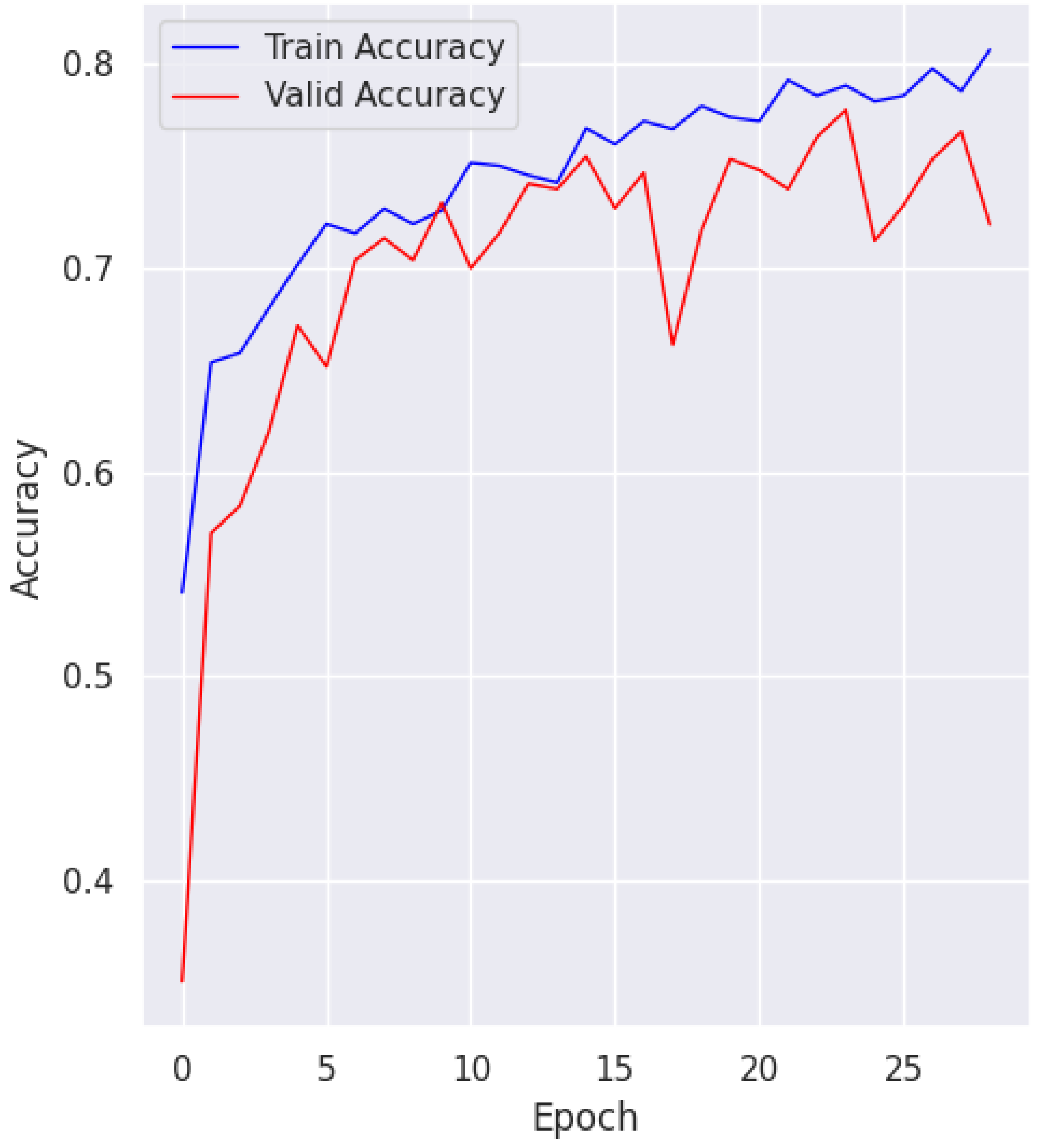
Accuracy: 0.8065

- **Validation**:

Loss: 0.876

Accuracy: 0.721

- **Test Loss:** 0.92894
- **Test Accuracy:** 0.7301



EVALUATION

Model 1 + data augmentation on segmented dataset

- **Training**:

Loss: 0.6827

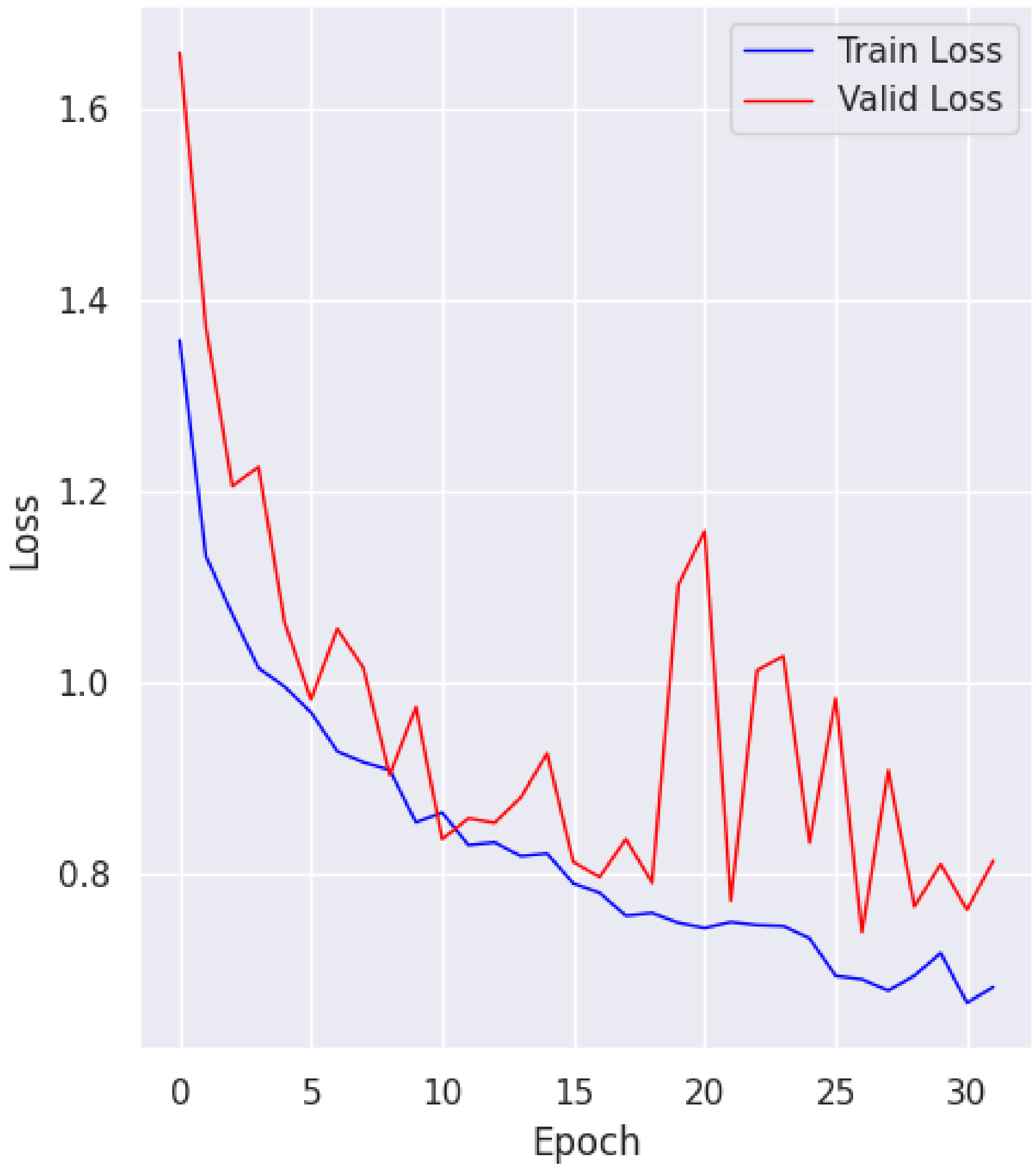
Accuracy: 0.7953

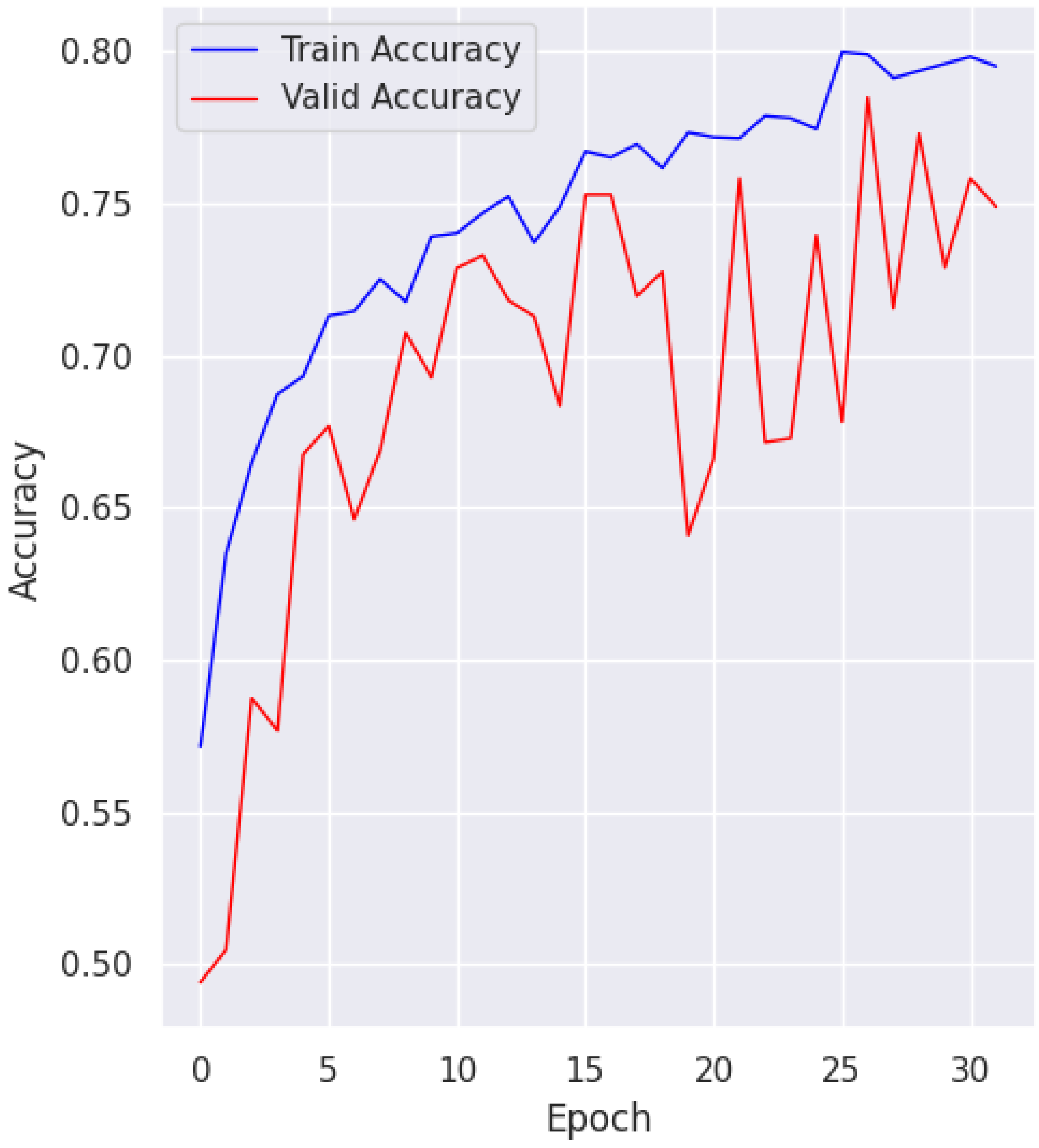
- **Validation**:

Loss: 0.8146

Accuracy: 0.7490

- **Test Loss**: 0.86612
- **Test Accuracy**: 0.753

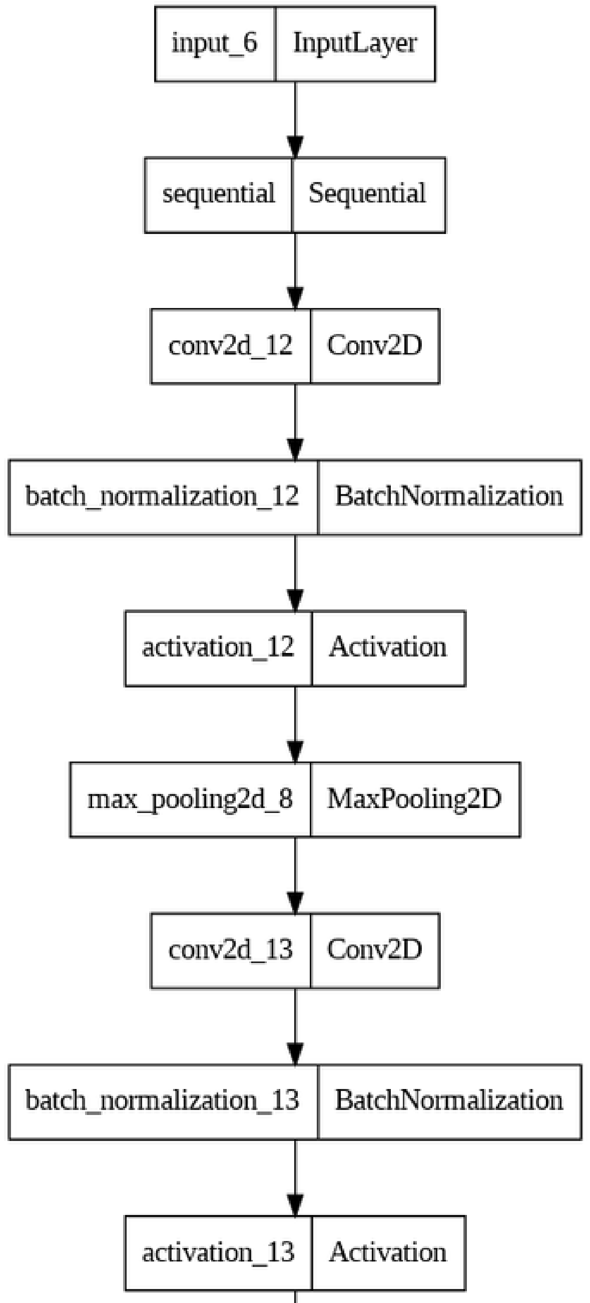


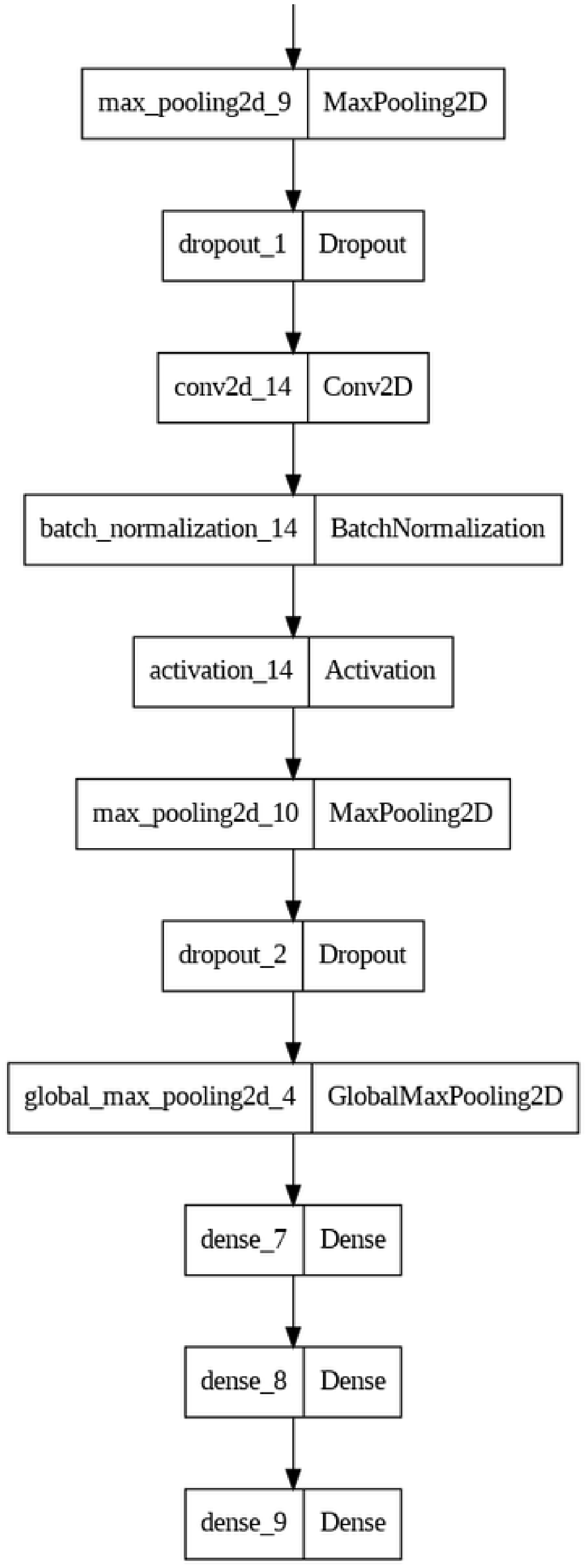


MODELS FROM SCRATCH

Model 2

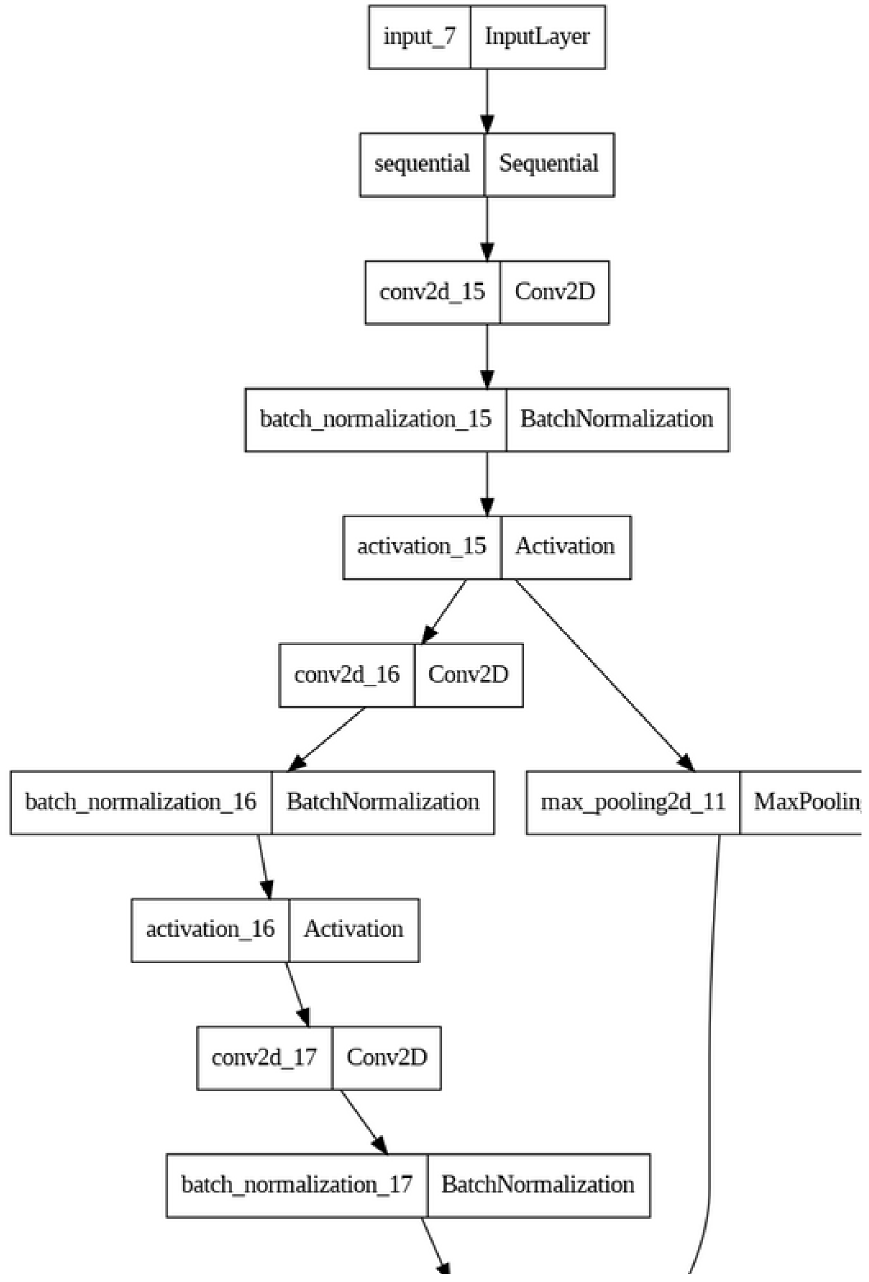
Total params: 161,669
Trainable params: 161,349
Non-trainable params: 320

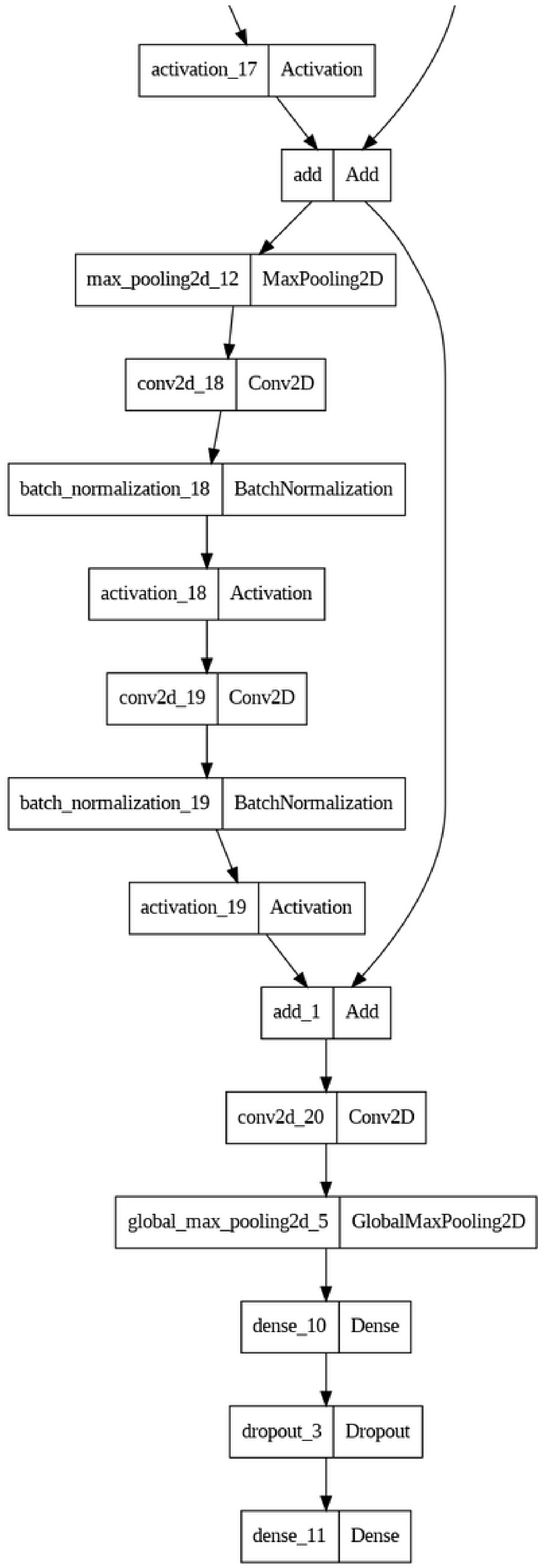




Model 3

Total params: 213,317
Trainable params: 212,677
Non-trainable params: 640





EVALUATION

Model 2

- **Test loss:** 0.96835
- **Test Accuracy:** 0.671

- **Training**:

Loss: 0.8069
Accuracy: 0.7520

- **Validation**:

Loss: 0.9234
Accuracy: 0.6889

Model 3

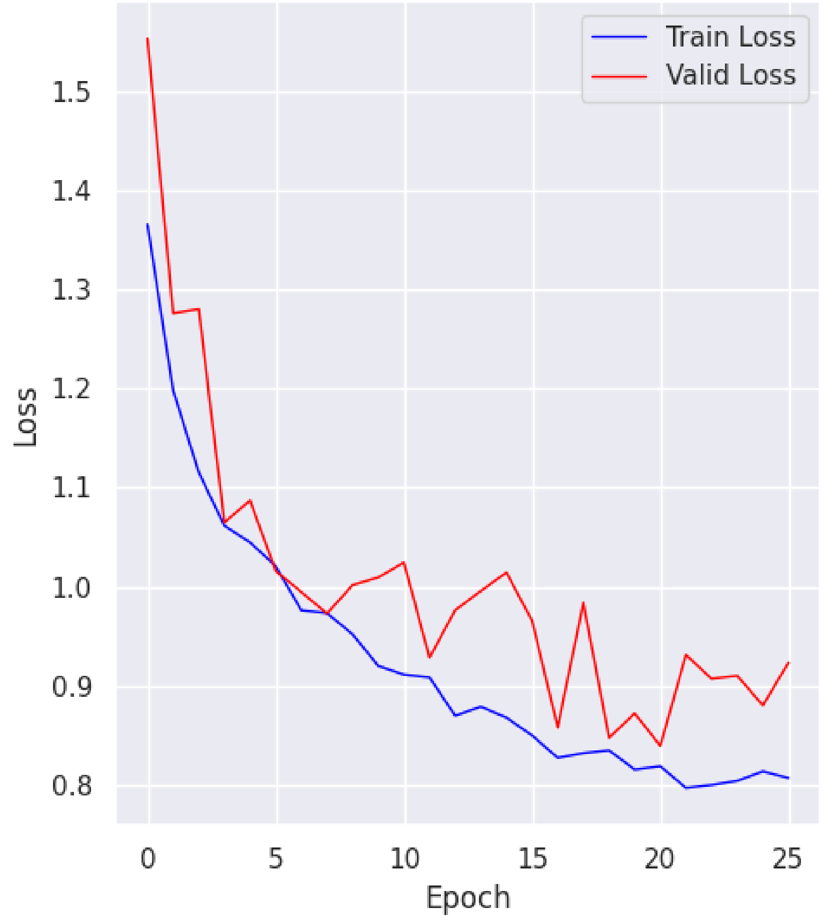
- **Test loss:** 0.86490
- **Test Accuracy:** 0.642

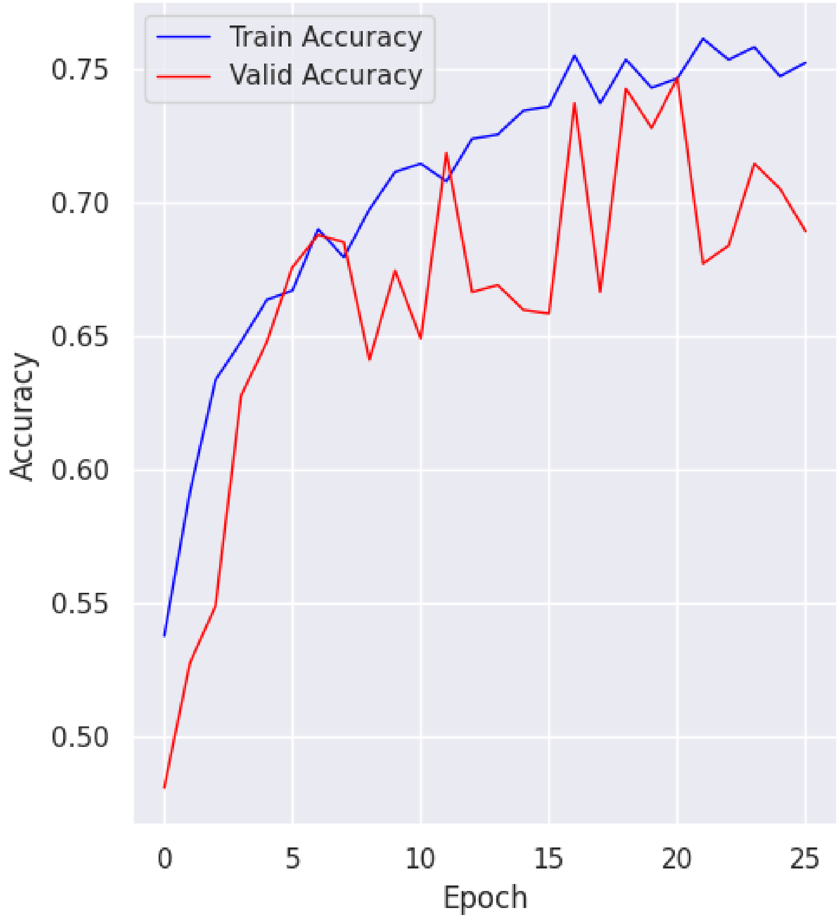
- **Training**:

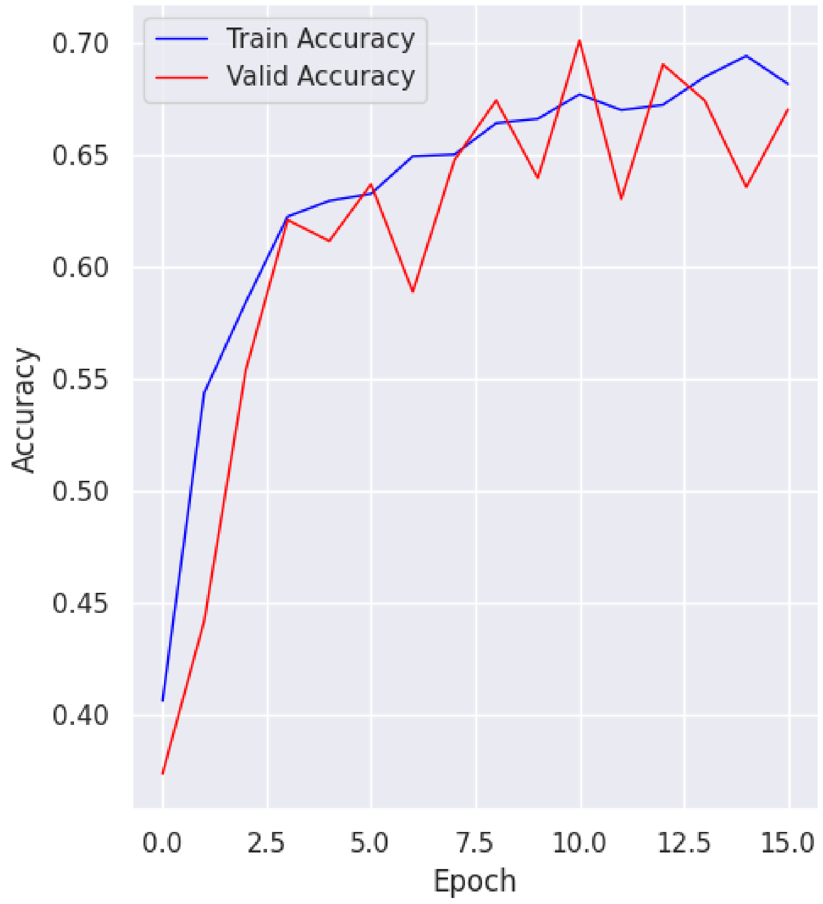
Loss: 0.8125
Accuracy: 0.6816

- **Validation**:

Loss: 0.8190
Accuracy: 0.6702







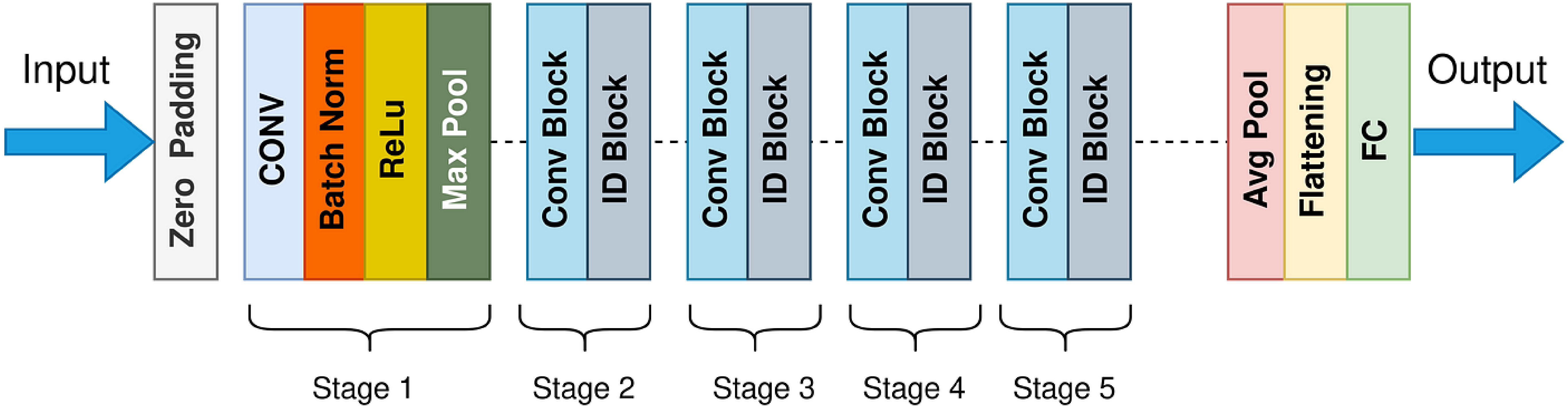
TRANSFER LEARNING

Transfer learning consists of taking features learned on one problem, and leveraging them on a new, similar problem.

We acted by following these steps:

1. Take layers from the ResNet50 pre-trained model
2. Freeze all layers except for the last block of ResNet50
3. Add some new layers
4. Train the new layers on our dataset
5. Lastly, train new layers on a generate dataset

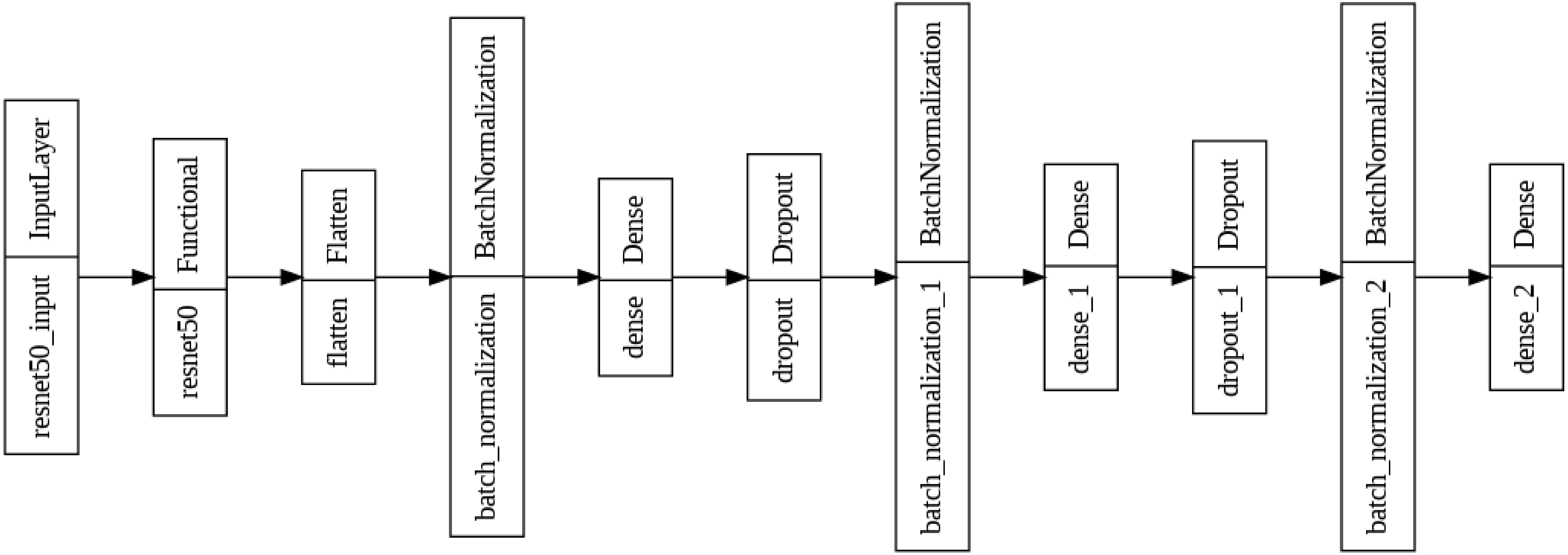
RESNET-50



The base model, ResNet-50, belongs to a family of models, where “50” represents the number of parameter layers in the architecture of network. In particular, it includes 48 convolutional layers, one Max Pool layer and one average Pool layer.

We freeze all layers except for the last block.

RESNET-50



InputLayer
resnet50_input
Functional
resnet50
Flatten
flatten
BatchNormalization
batch_normalization
Dense
dense
Dropout
dropout
BatchNormalization
batch_normalization_1
Dense
dense_1
Dropout
dropout_1
BatchNormalization
batch_normalization_2
Dense
dense_2

EVALUATION

- **Training**:

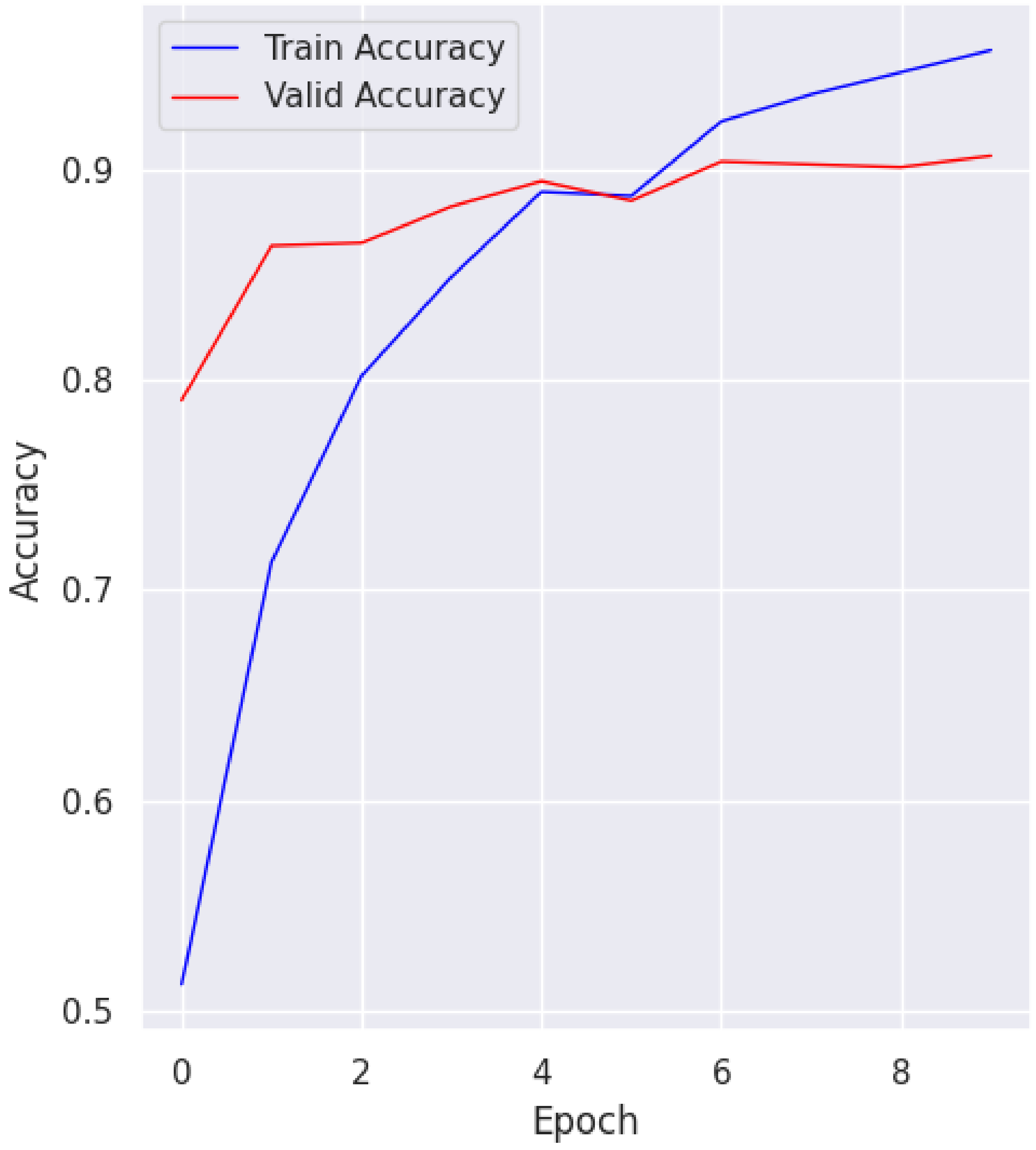
Loss: 0.2056

Accuracy: 0.9568

- **Validation**:

Loss: 0.2776

Accuracy: 0.9065



EVALUATION

- **Training**:

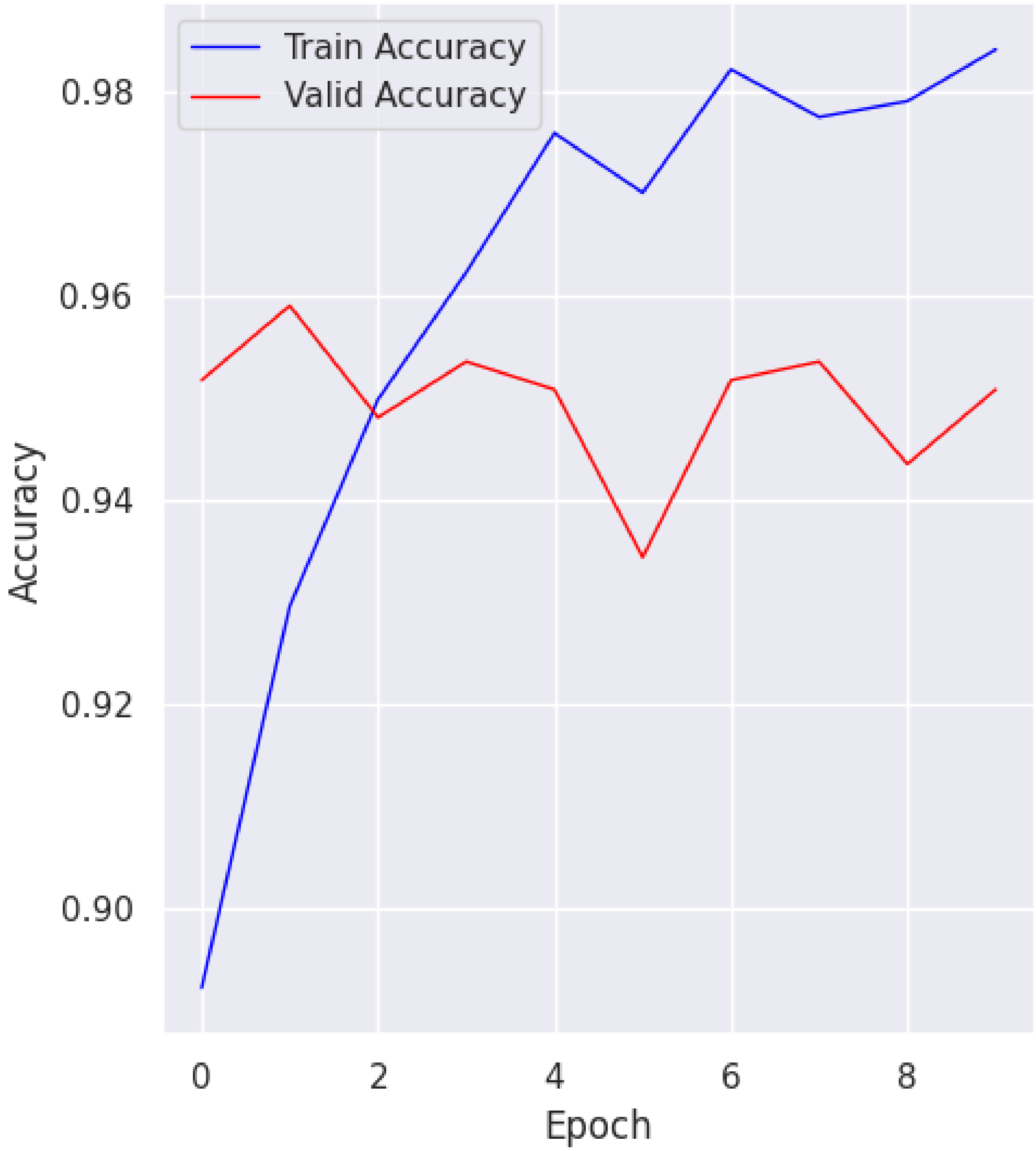
Loss: 0.1044

Accuracy: 0.9841

- **Validation**:

Loss: 0.1516

Accuracy: 0.9508



FINE-TUNING

Fine-Tuning pre-trained models is a very powerful training technique that is used to re-purpose a model trained on the ImageNet dataset for use with a custom dataset. The goal of fine-tuning is to allow a portion of the pre-trained layers to retrain.

We acted by following these steps:

1. Remove the fully connected nodes at the end of the network
2. Replace the fully connected nodes with the initialized ones
3. Freeze the initial layers in the convolutional base
4. Start training

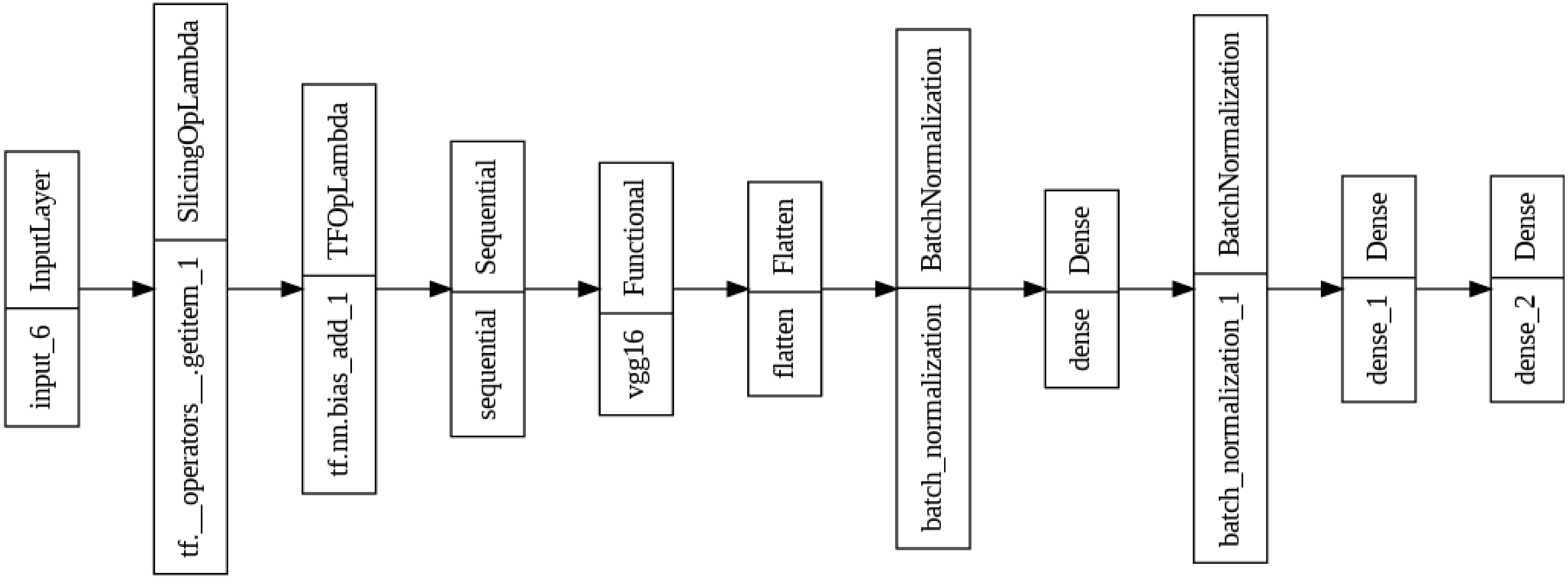
VGG-16

VGG-16 is one of the CNN architecture which is considered as very good model for Image classification, in fact is trained on 1.2 million images to classify 1000 different categories.

The VGG16 Model has 16 Convolutional and Max Pooling layers, 3 Dense layers for the Fully-Connected layer, and an output layer of 1,000 nodes.

We fine tune the last 4 convolutional layers.

VGG-16



input_6
InputLayer
tf.__operators__.getitem_1
SlicingOpLambda
tf.nn.bias_add_1
TFOpLambda
sequential
Sequential
vgg16
Functional
flatten
Flatten
batch_normalization
BatchNormalization
dense
Dense
batch_normalization_1
BatchNormalization
dense_1
Dense
dense_2
Dense

EVALUATION

- **Training**:

Loss: 0.1548

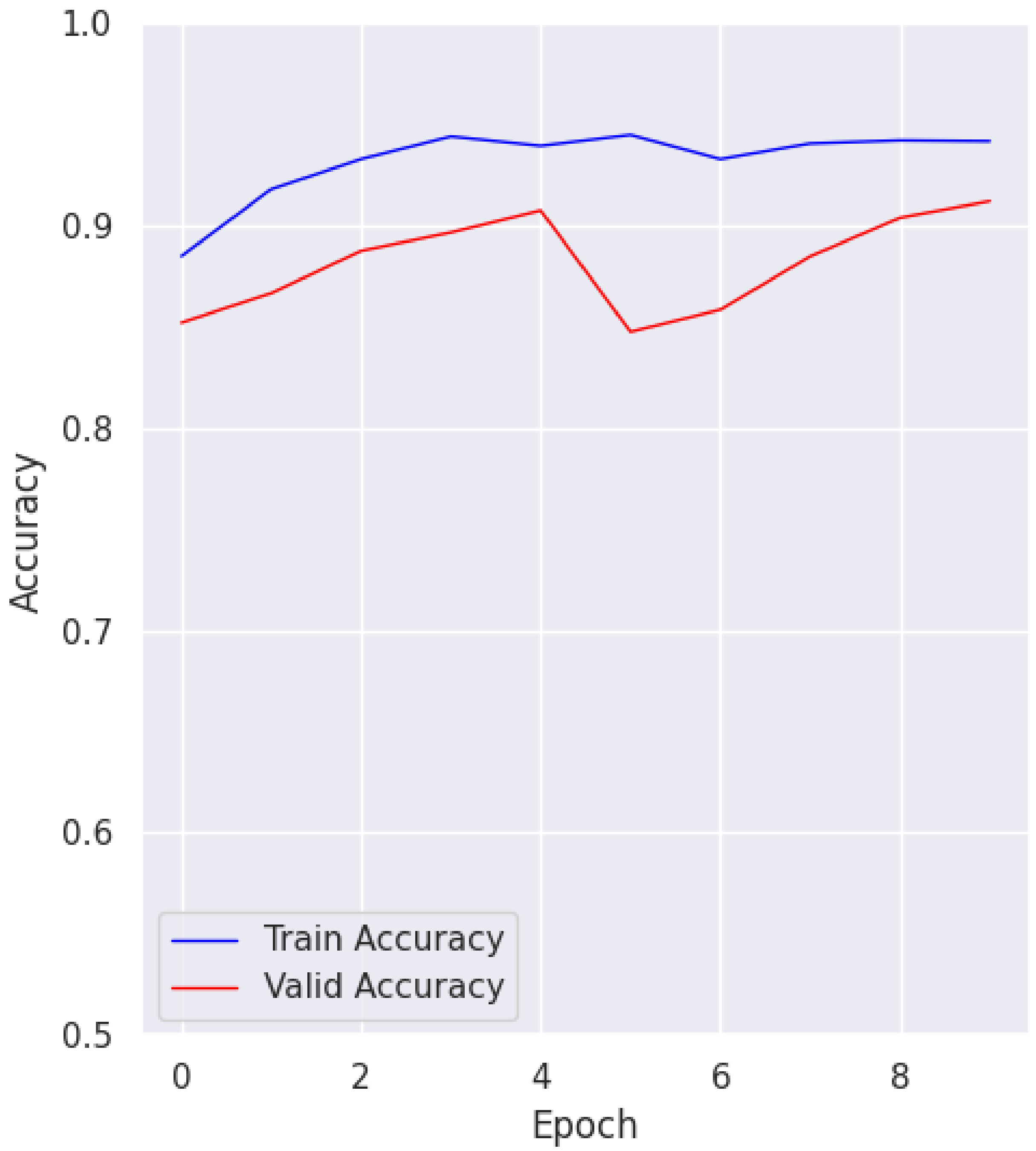
Accuracy: 0.9416

- **Validation**:

Loss: 0.3211

Accuracy: 0.9119

- **Test loss:** 0.35881
- **Test Accuracy:** 0.878



CONCLUSION

We obtained the best classification results through the fine tuning approach on the VGG16 model with validation accuracy 91% and 88% accuracy on test set.

But it should be noted that the ResNet50, adopted by us, has good results as well, with validation accuracy 90%.

The best model from scratch developed by us is Model 1 trained on segmented data, with the accuracy on the test set 75%.

The same model architecture but trained on original data has accuracy on test set 73%.

THANKS FOR YOUR ATTENTION!