Aurora Cerabolini

Andrea Malinverno

Foundations of Deep Learning

Concrete Crack Images for Classification

Concrete Crack Images for Classification

The dataset contains concrete images having cracks and it is divided in two classes called Positive and Negative. Images having cracks belong to the class «Positive» and the images without cracks belong to the class «Negative». Each class contains 20000 images, so in total there are 40000 images.

We want to build a Convolutional Neural Network to correctly classify the images belonging to the Positive class and the images belonging to the Negative one.

Data Preparation

To achieve our goal, we have split our dataset into three sets – Training, Validation and Test – using the stratified sampling technique. We have used the 70% of total data for the Training set, the 20% for the Validation set and the 10% for the Test set.

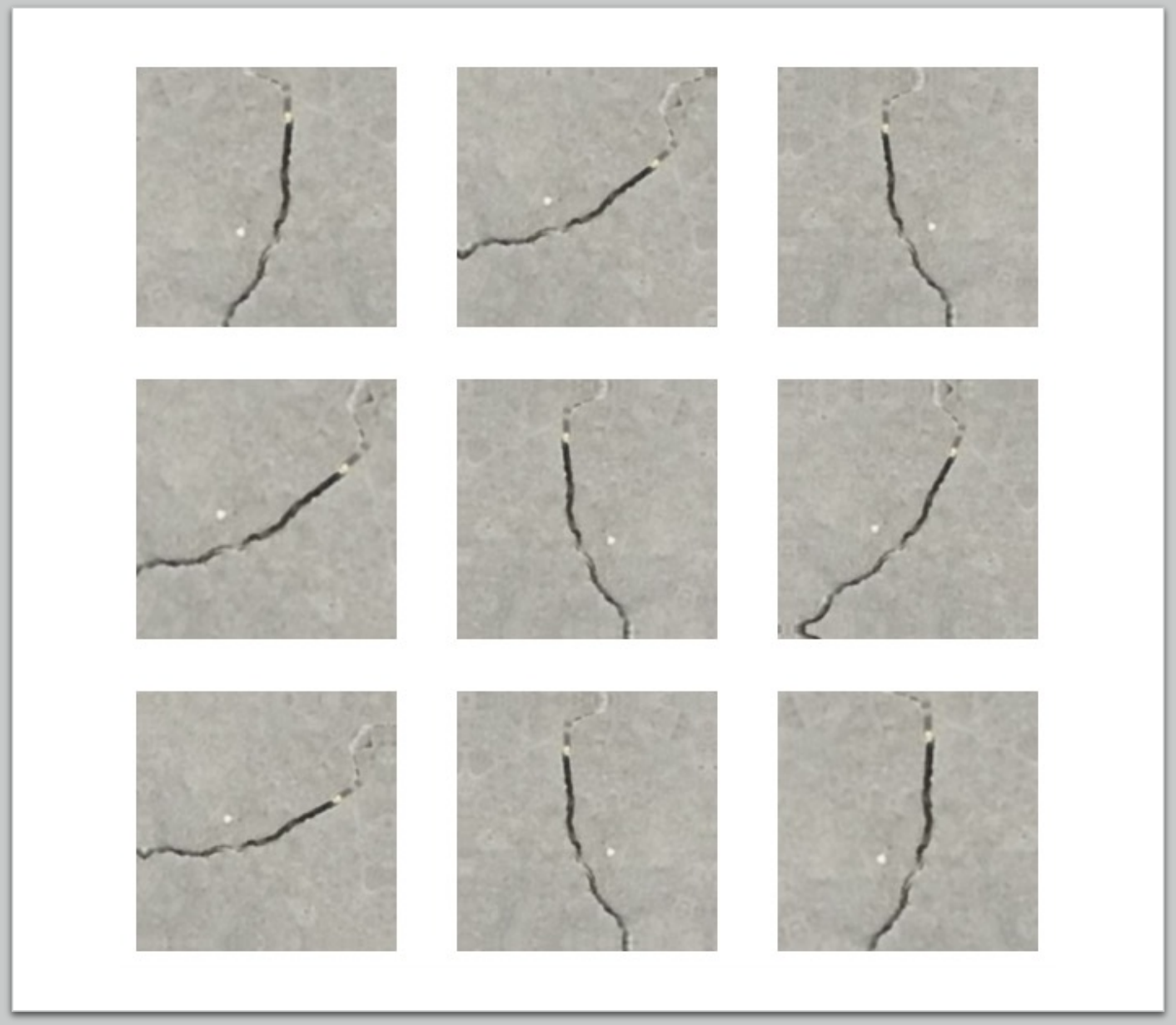
We have used the Training Set to train the model and the Validation set to evaluate the loss and the accuracy of the model at the end of each epoch of training. The Test set has been used to evaluate the already trained model.

We have set the batch size to 32 and this term refers to the number of training examples utilized in one training iteration.
Since all images within the batch must be the same size, we have resized all images to (224, 224).

Image Data Augmentation

We have generated additional training data from our existing images using random transformations that produce believable-looking images. In this way our model is exposed to more aspects of the data so it can generalize better.

In particular, we have applied horizontal flip, random rotation and random zoom to each image.



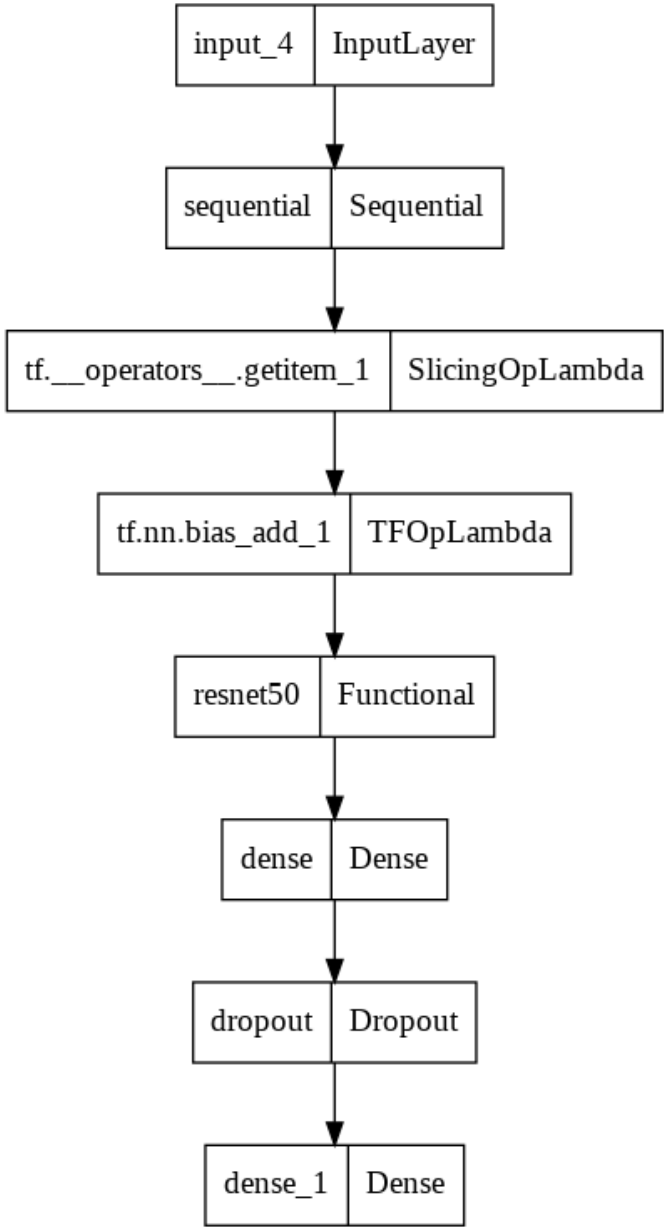
Our solution

We have implemented a Convolutional Neural Network based on Transfer Learning.

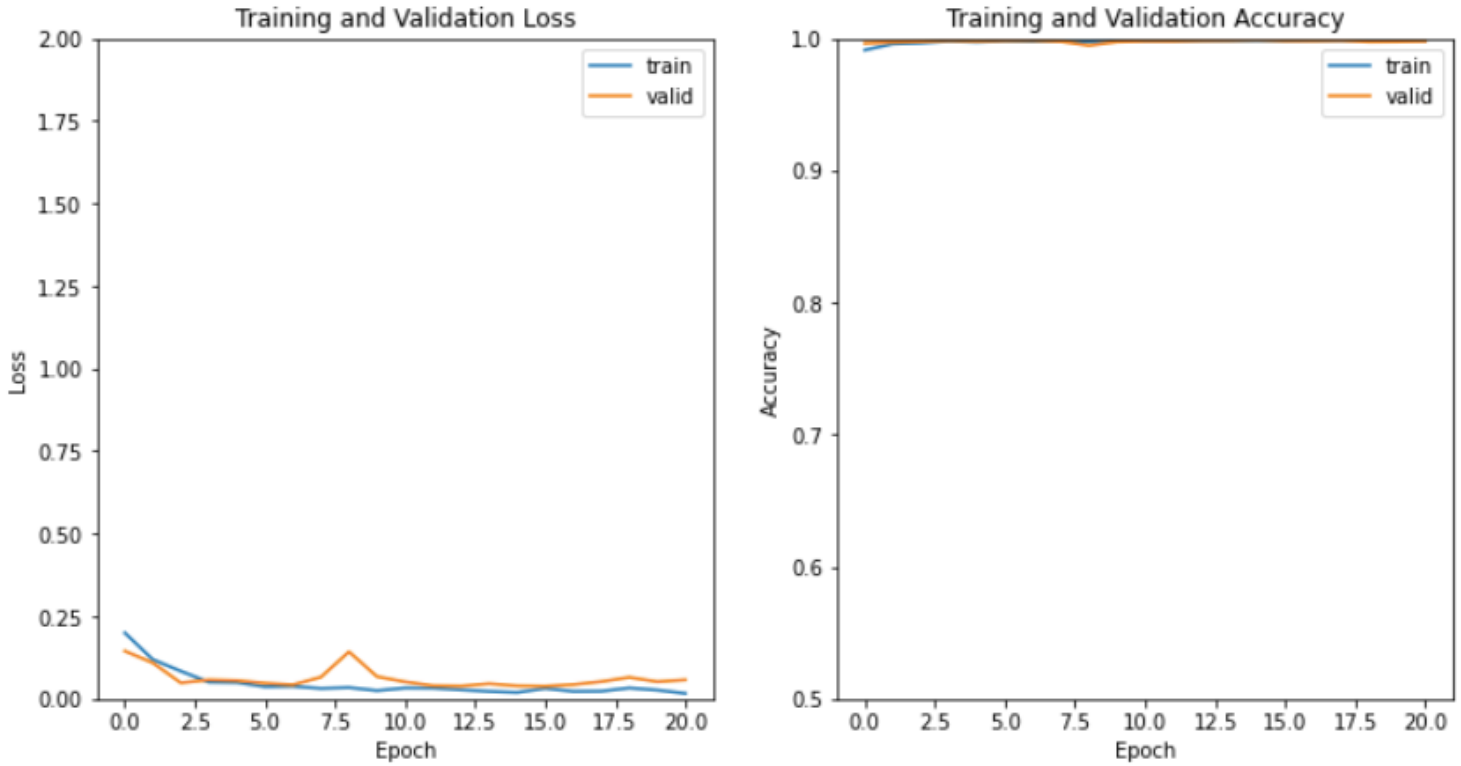
Our base model has been created from the ResNet 50 model that is pre-trained on the ImageNet dataset, a large dataset consisting of 1.4M images and 1000 classes. We have also applied to our input images the same preprocessing as the one which was used during the training of ResNet 50.

Our final model has been built by chaining together the data augmentation, the preprocessing layer, the base model, a fully connected layer with ReLU activation, a dropout layer and a final fully connected layer with softmax activation.

To compile the model we have used the RMSprop optimizer and the categorical crossentropy as loss function.
To train the model we have used early stopping as a form of regularization to avoid overfitting: after 5 epochs of no improvement (or even getting worse) of the validation loss curve, the training is stopped.
To evaluate the model's performance we have used the accuracy metric.



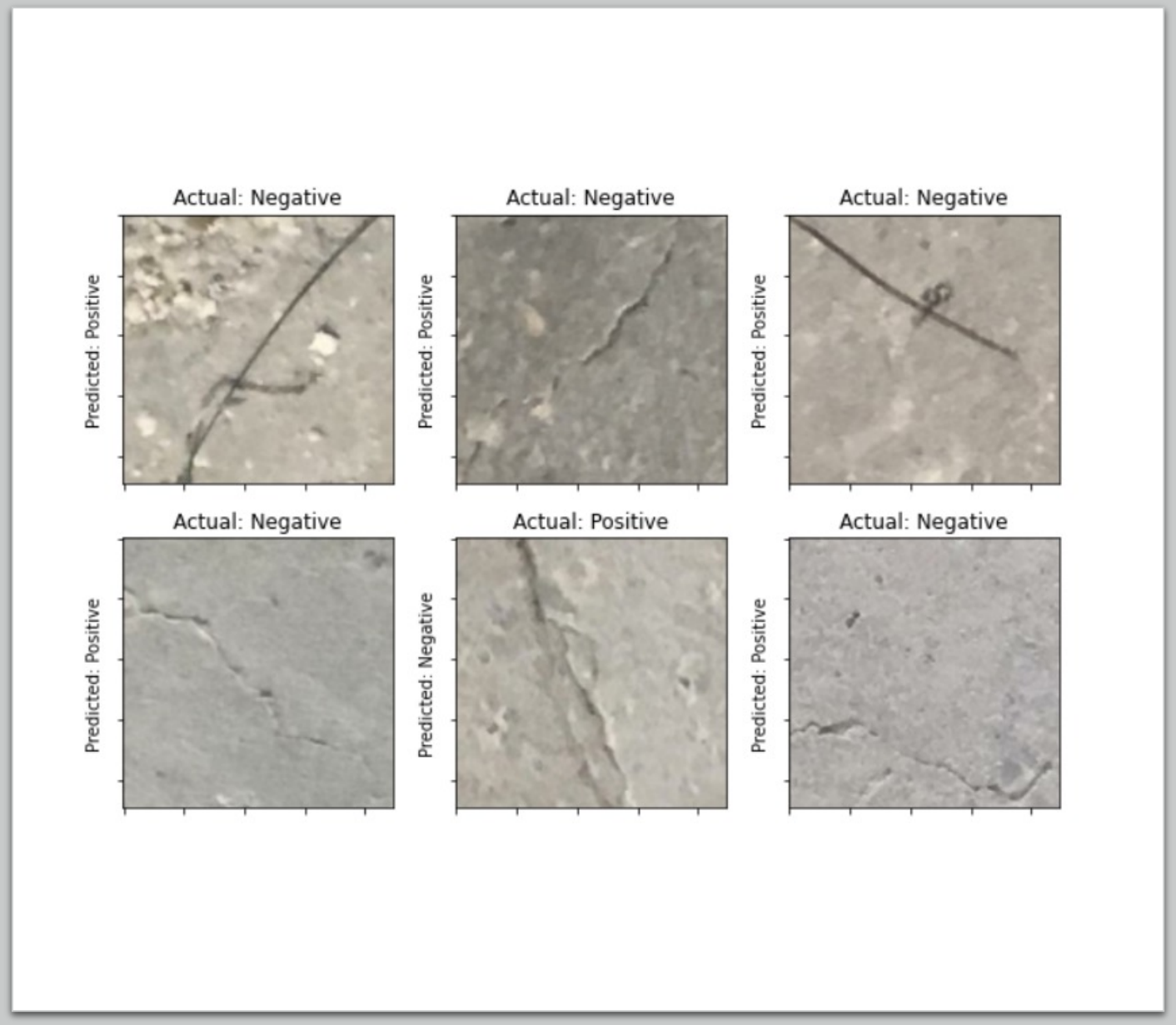
The Train and Validation Loss Curve and the Train and Validation Accuracy Curve are shown below:



The model training time lasted 45 minutes. We have set 40 training epochs but the early stopping has ended training at 21 epochs

Evaluation

We have evaluated our model on the test set, obtaining a loss of 0.0156 and an accuracy of 0.9985: this model misclassifies only 6 images out of a total of 4000.

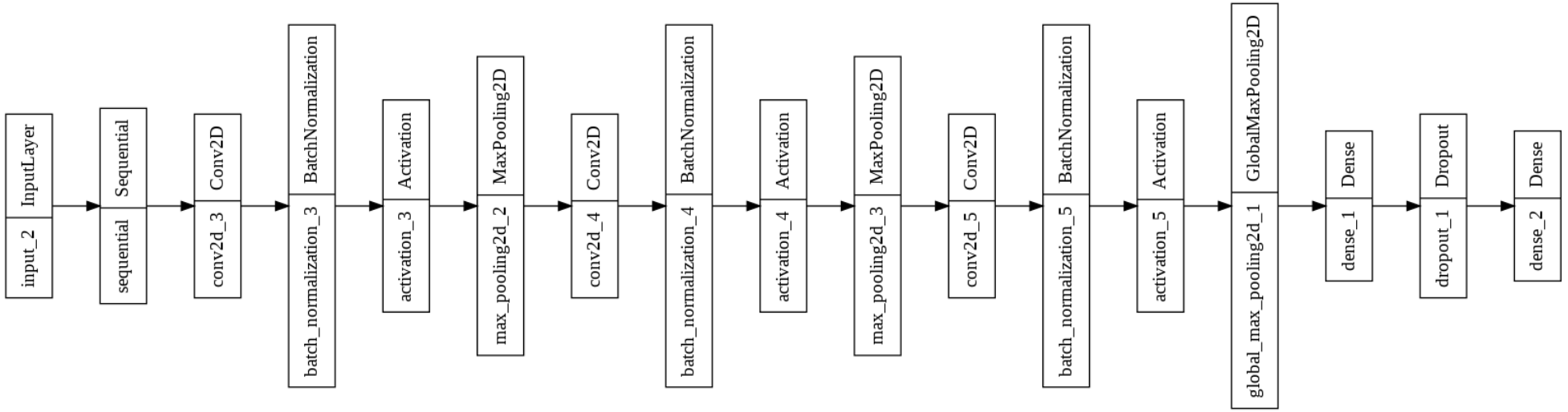


Alternative solutions

In the same way as we did with our solution, also for all the architectures of our alternative models we have always used the ReLU activation function, except for the last fully connected layer where we have used the softmax activation. To compile the models we have used the RMSprop optimizer and the categorical crossentropy as loss function. To train them we have used early stopping as a form of regularization to avoid overfitting; to evaluate their performances we have used the accuracy metric.

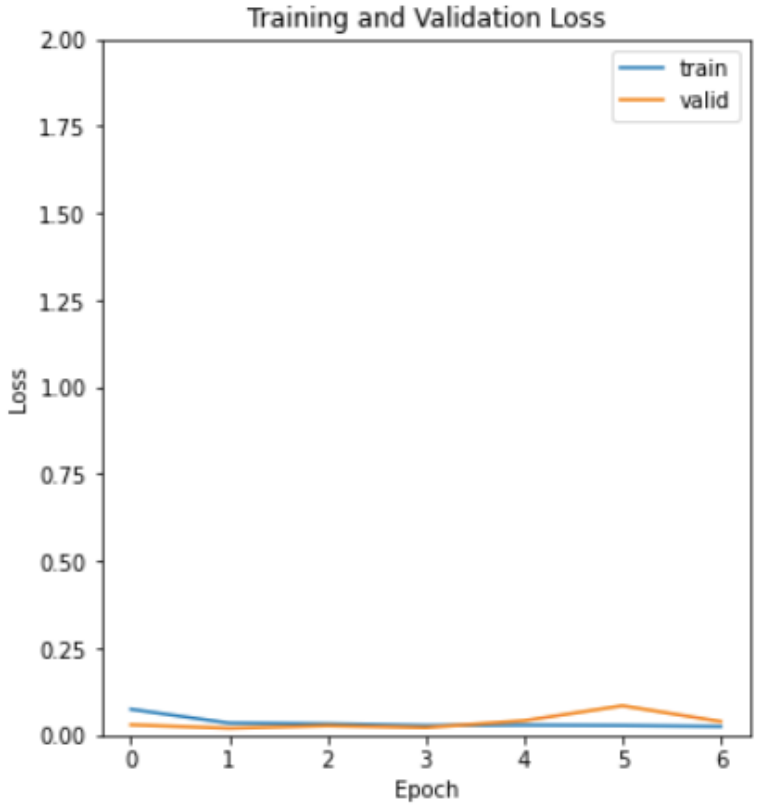
- First Model

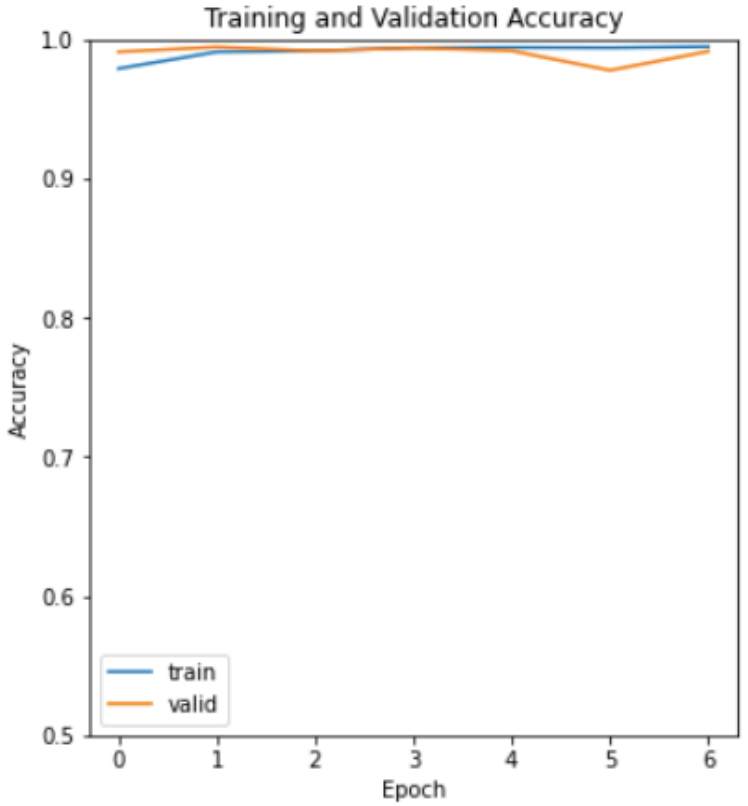
In the first 2D convolution layer we have set the number of filter to 32 with dimensionality 3x3; on each following convolution layer we have doubled every time the number of filters.



The Train and Validation Loss Curve and the Train and Validation Accuracy Curve are shown on the side.

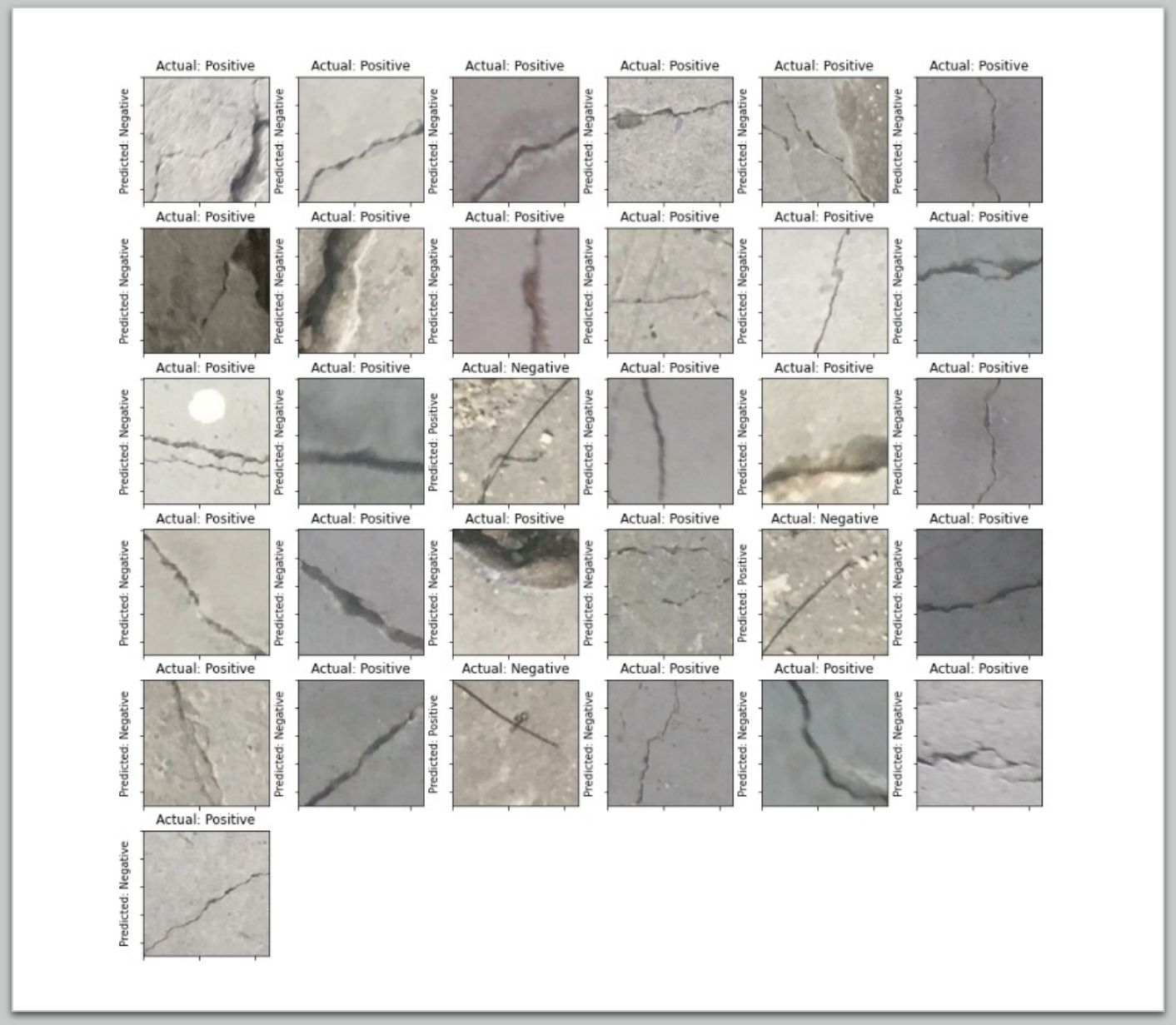
The model training time lasted 7 minutes. We have set 40 training epochs but the early stopping has ended training at 7 epochs





Evaluation of the first model

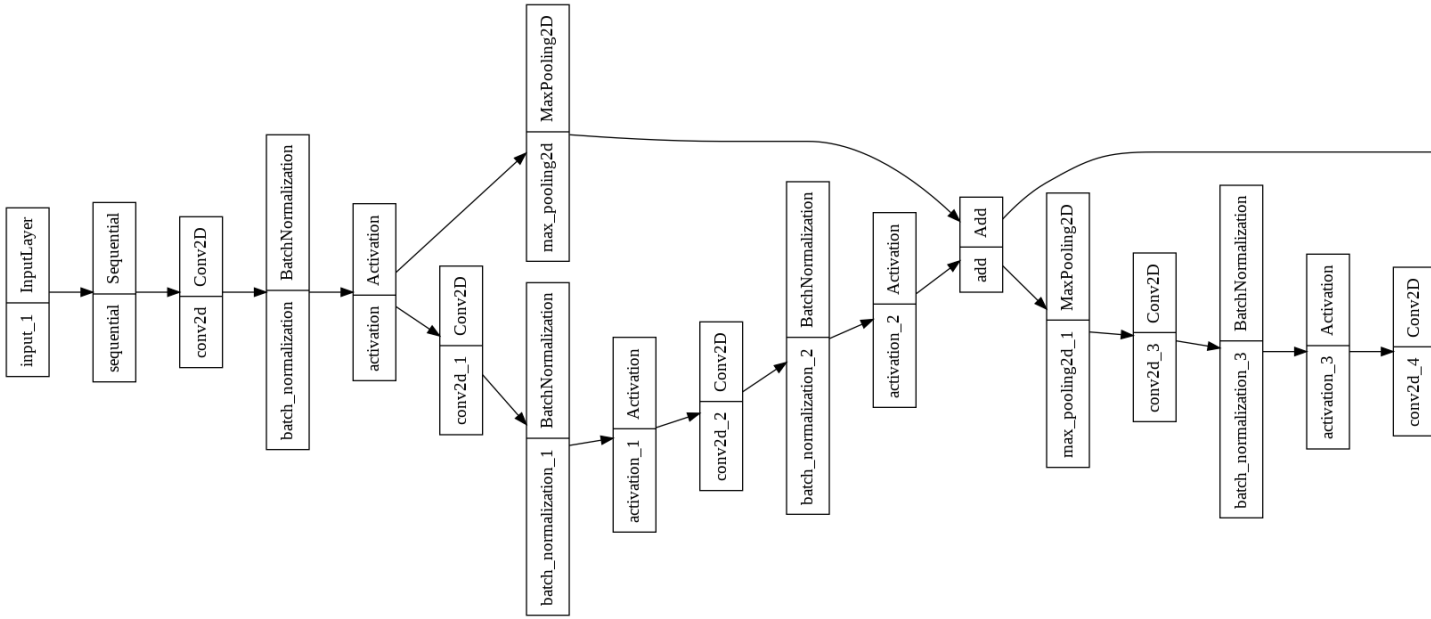
We have evaluated this first model on the test set, obtaining a loss of 0.0327 and an accuracy of 0.9923: this model misclassifies 31 images out of a total of 4000.



We have also built a second and a third Convolutional Neural Networks increasing the number of layers and using residual blocks. A residual block is a stack of layers set in such a way that the output of a layer is taken and added to another layer deeper in the block.

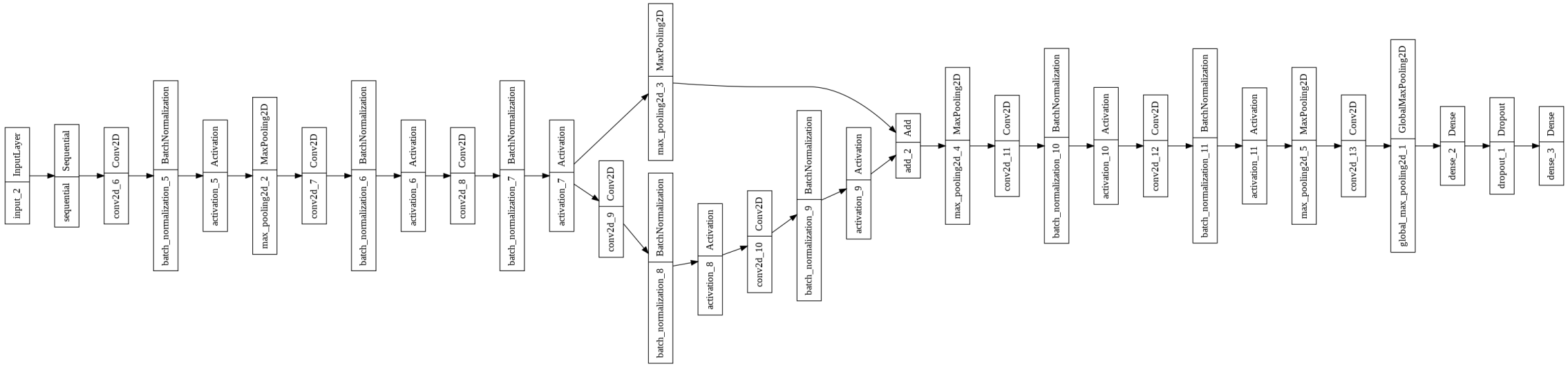
- Second Model

In each 2D convolution layer we have set the number of filter to 64: the first 64 filters have a dimensionality of 7x7 and the other ones have a dimensionality of 3x3.



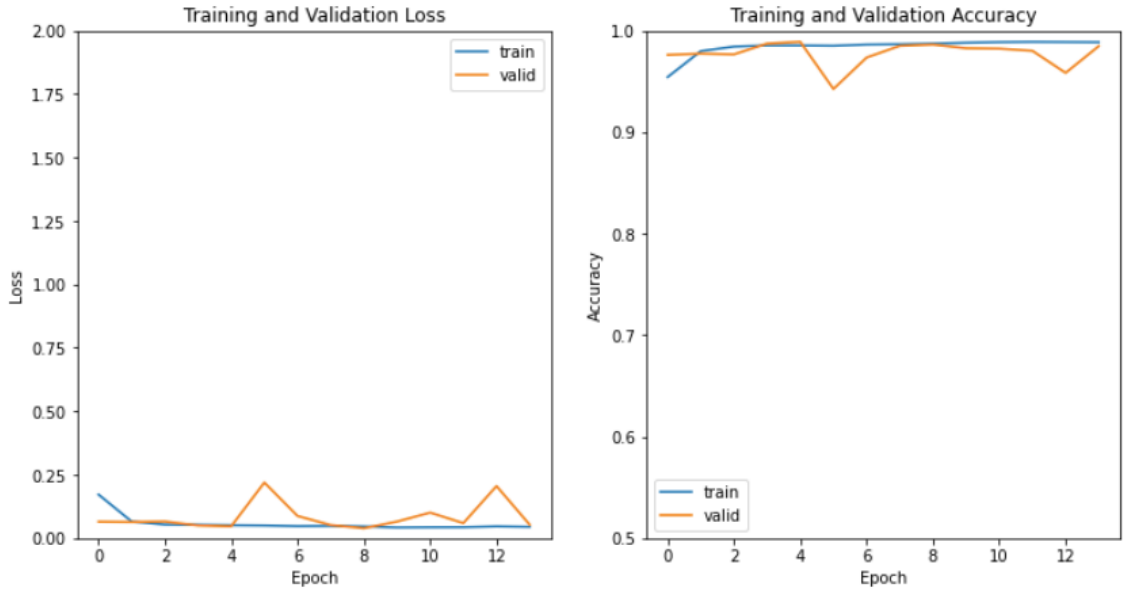
- Third Model

In the first 2D convolution layer we have set the number of filter to 32 with a dimensionality of 7x7; in the following four convolution layers the number of filters was set to 64 with a dimensionality of 3x3, then we have added two convolution layers with the number of filters set to 128 (dimensionality 3x3) and in the last convolution layer the number of filter was set to 64 with a dimensionality of 3x3.



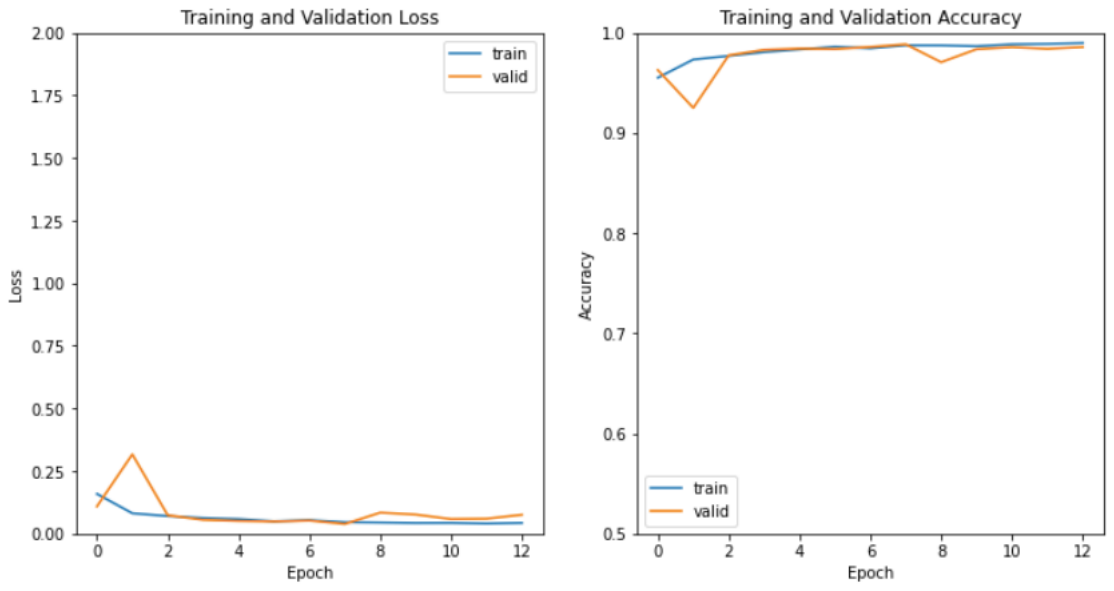
The Train and Validation Loss Curve and the Train and Validation Accuracy Curve for this two models are shown below:

Second Model



The second model training time lasted 2 hours and 47 minutes. We have set 40 training epochs but the early stopping has ended training at 14 epochs

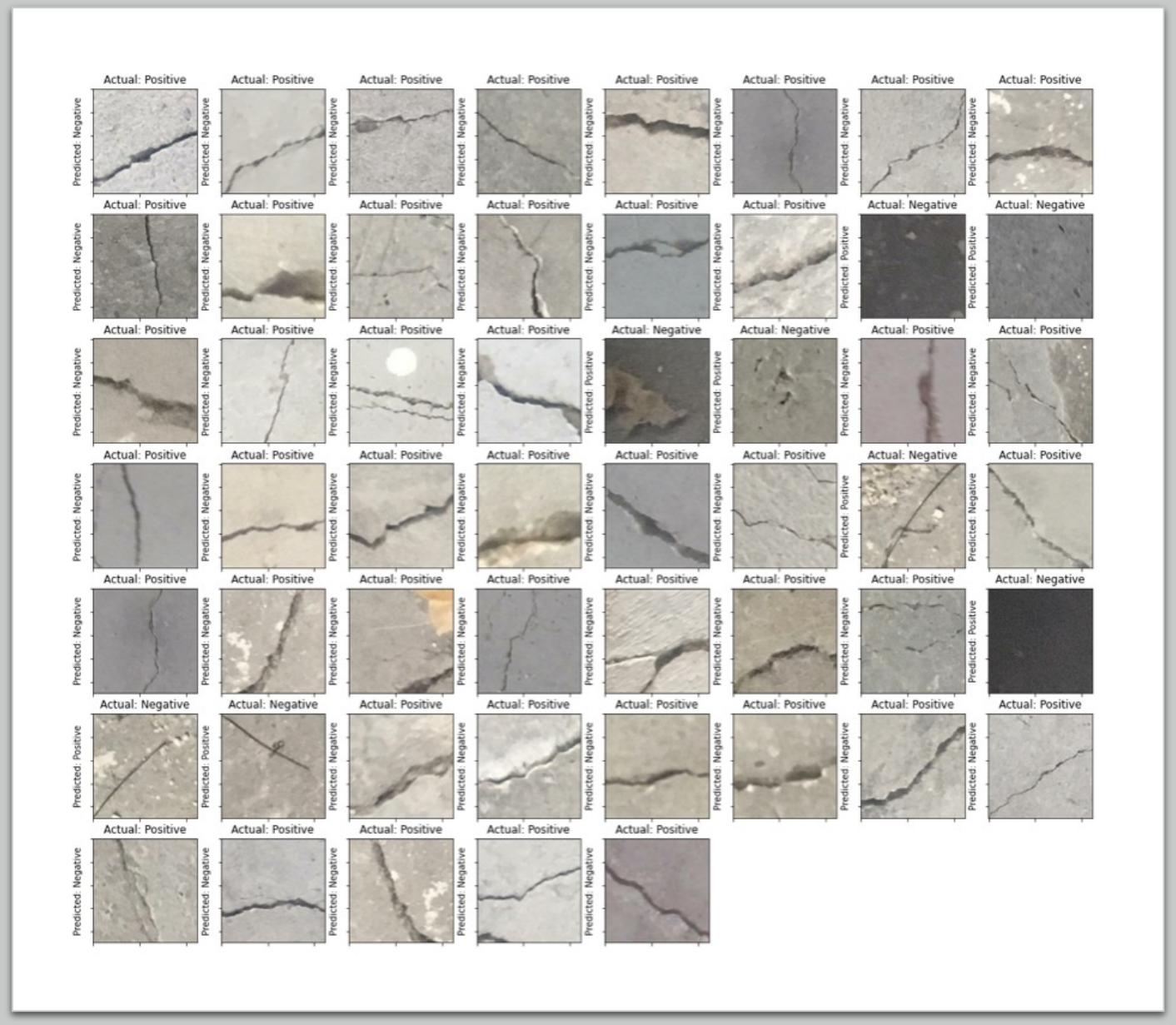
Third Model



The third model training time lasted 4 hours and 30 minutes. We have set 40 training epochs but the early stopping has ended training at 13 epochs

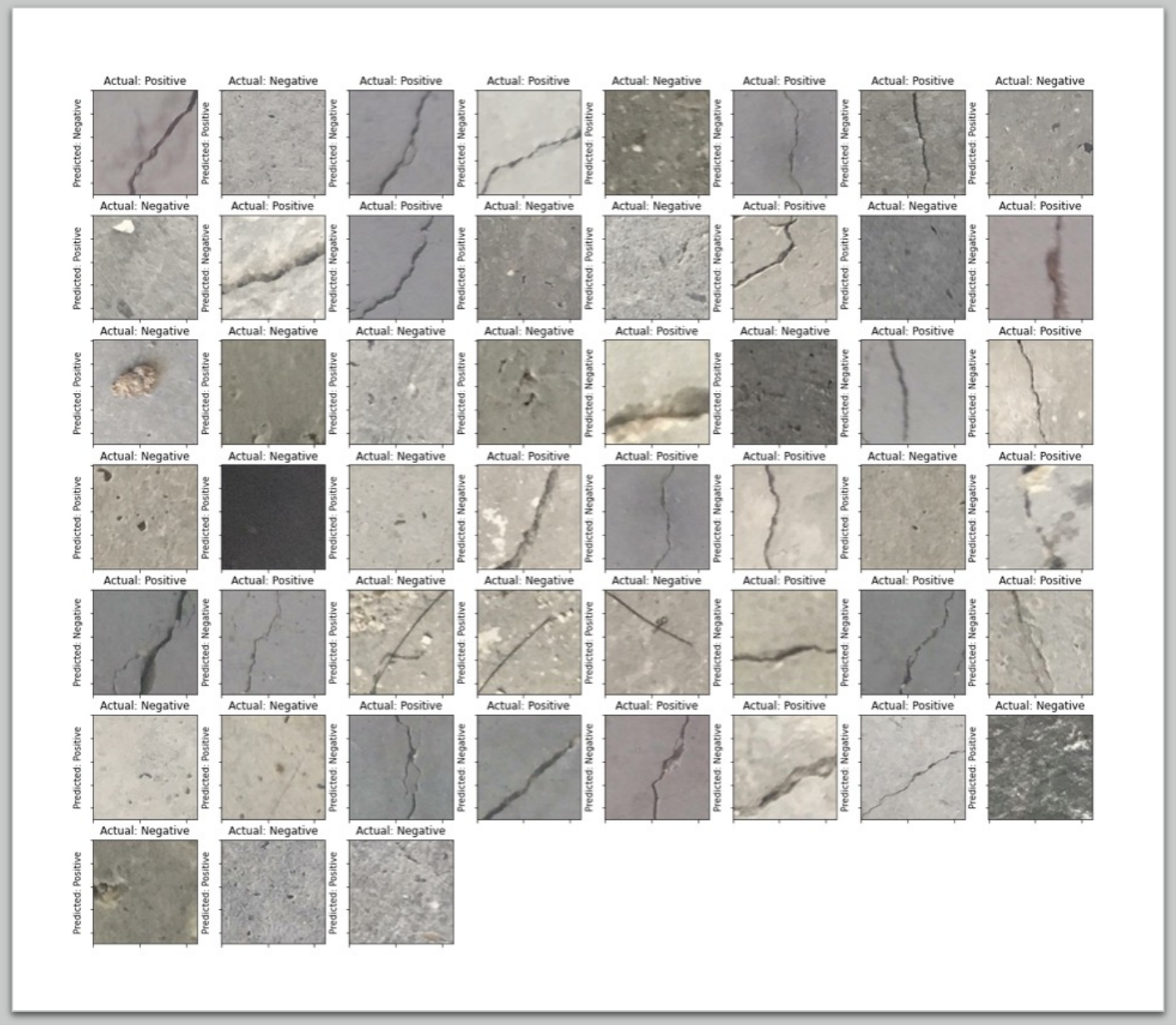
Evaluation of the second model

We have evaluated the second model on the test set, obtaining a loss of 0.0426 and an accuracy of 0.9868: this model misclassifies 53 images out of a total of 4000.



Evaluation of the third model

We have evaluated the third model on the test set, obtaining a loss of 0.0602 and an accuracy of 0.9872: this model misclassifies 51 images out of a total of 4000.

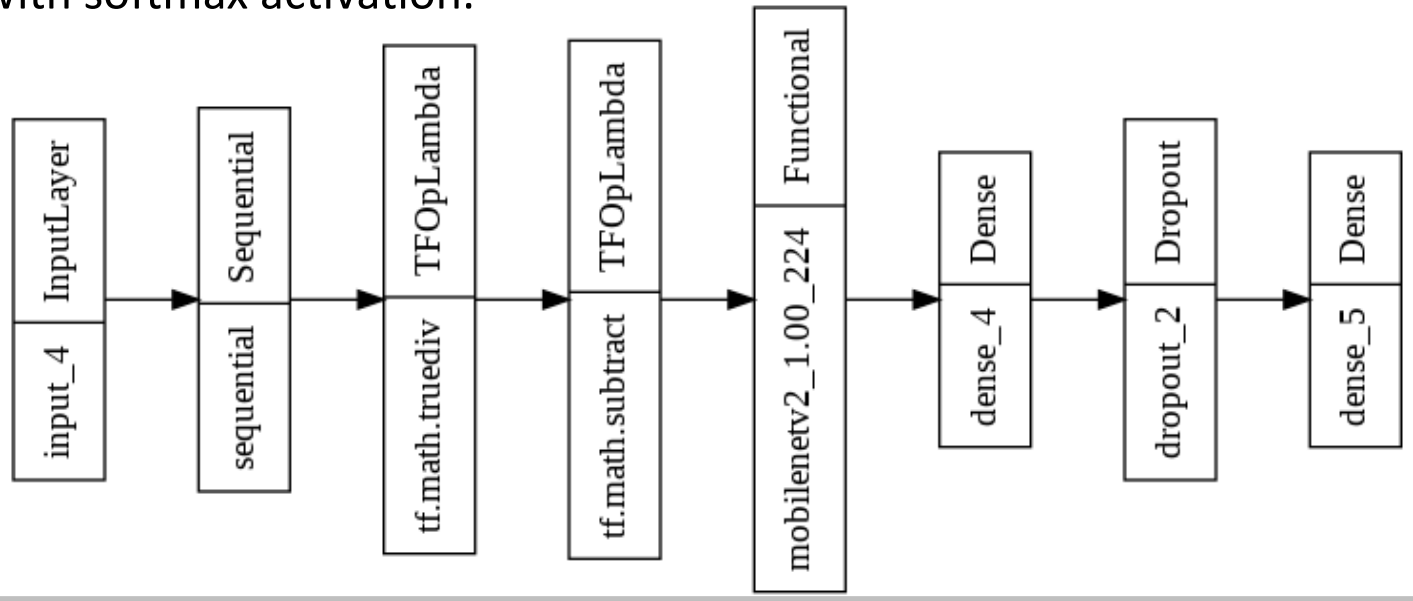


- Fourth Model

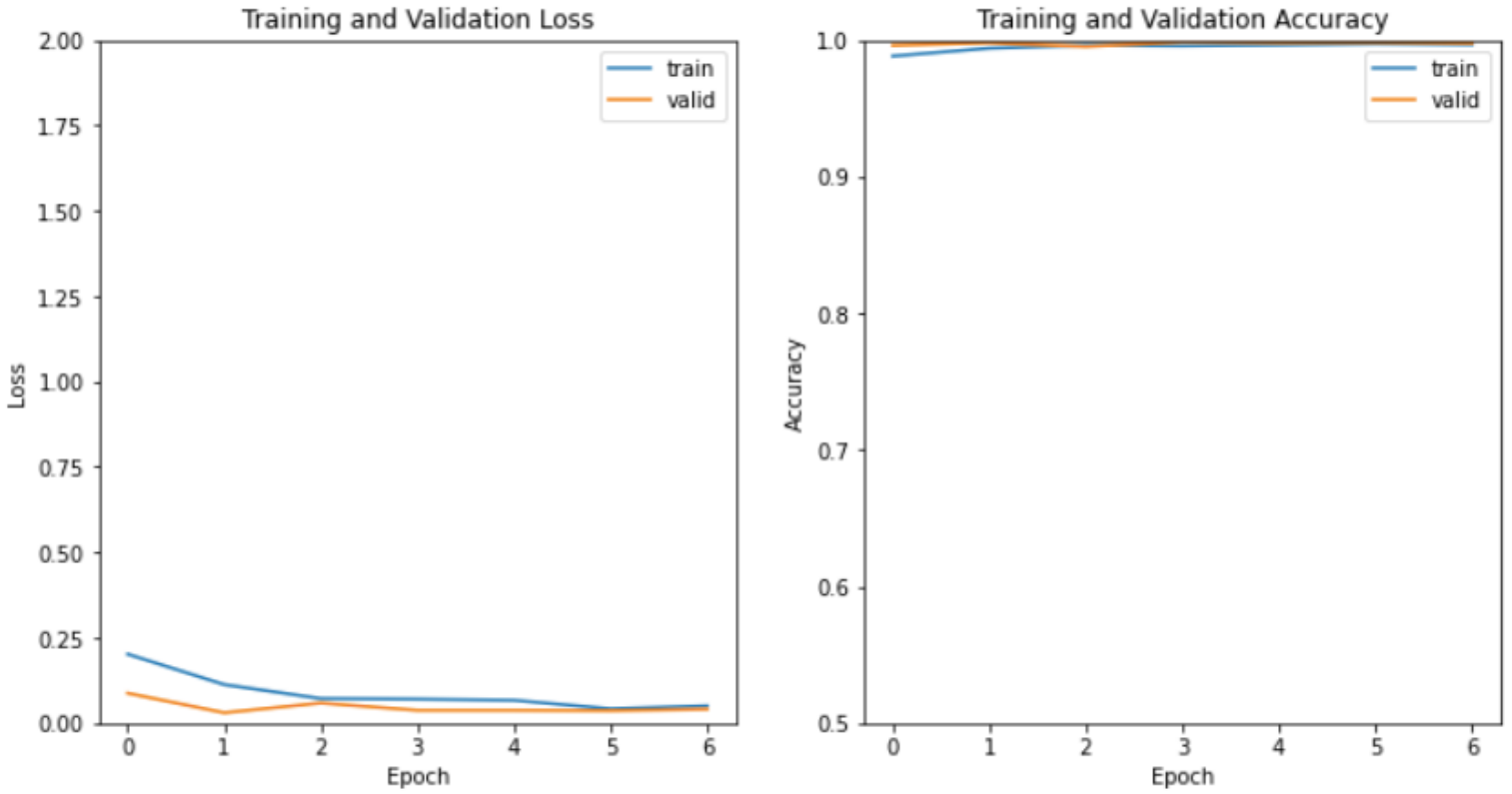
The last Convolutional Neural Network we have implemented is based on Transfer Learning.

Our base model has been created from the MobileNet V2 model that is pre-trained on the ImageNet dataset. We have also applied to our input images the same preprocessing as the one which was used during the training of MobileNet V2, in the same way as we did for our solution.

The final model has been built by chaining together the data augmentation, the preprocessing layer, the base model, a fully connected layer with ReLU activation, a dropout layer and a final fully connected layer with softmax activation.



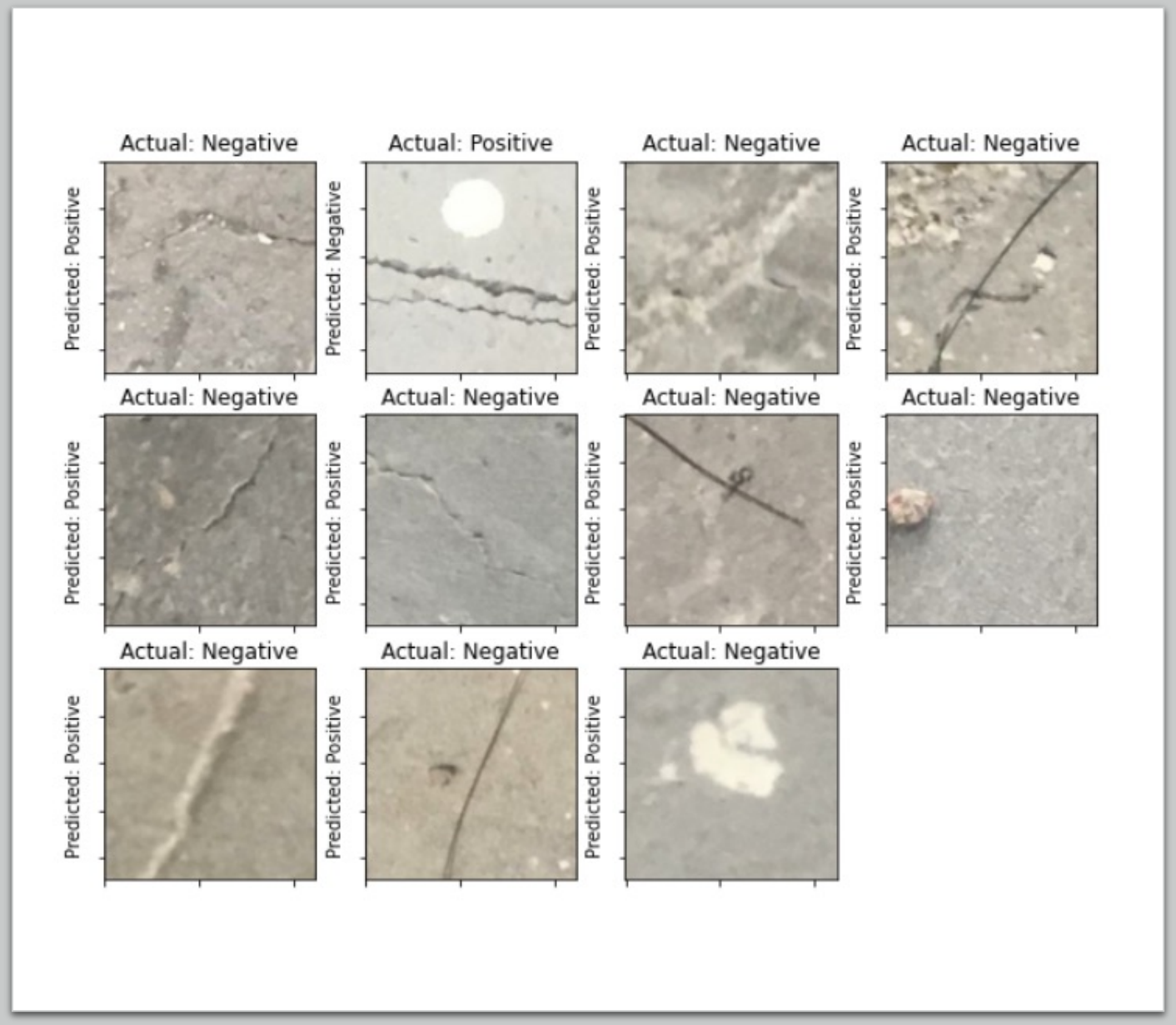
The Train and Validation Loss Curve and the Train and Validation Accuracy Curve are shown below:



This model training time lasted 7 minutes. We have set 40 training epochs but the early stopping has ended training at 7 epochs

Evaluation of the fourth model

We have evaluated this last model on the test set, obtaining a loss of 0.0566 and an accuracy of 0.9973: this model misclassifies 11 images out of a total of 4000.



Conclusion

Among all the models implemented, the best one turned out to be the Convolutional Neural Network based on Transfer Learning with the base model created from the ResNet 50.

This model not only has a high test accuracy but, considering both the accuracy and the training time jointly, this model turns out to have the best accuracy-training time ratio.