Maastricht University

Deep Learning

Computer Vision

I6278383 - Pierre Onghena
I6255812 - Niklas Belo

1 Abstract

Emotion Recognition is a machine learning task, in which the emotion of a human gets determined, based on input like sound, video or text. In this report, the focus is on interpreting facial features to predict the emotion of a human. In order to gather insights about the properties and optimization techniques, multiple experiments will be conducted through the process of trying different training parameter values. Herewith, parameter sweeps are driven to observe a good set of neural network weight values. Furthermore, a discussion can be established in the evaluation of the composed architectures and their corresponding performance in regard to the Facial Expression Recognition 2013 data set.

2 Introduction

This report performs a benchmark analysis of the convolutional neural network concerning the setting of various configurations and architectures. The provided data set was categorized according to their purpose of being a training or test set. In addition, the pixels of the training set are represented according to the image size of (48, 48) and normalized for conducting constructive experiments. For training and testing, the private training set which consists of 28709 labelled images was retained to shuffle and split it with respective ratio of (70% / 30%). The final verification was performed on the public and private test sets that were categorized by the provided data set.

Besides a seeking for the optimal architecture consisting of several layers, the purpose of the operations after each convolutional layer will be discussed with different parameter settings. By intuition, the hyperparameter which controls the learning rate is initially set at 0.005 for configuring the neural network as it positively converges the optimization process. Moreover, a kernel filter with size (3x3) and the zero-padding technique were applied to the convolutional layers in order to preserve the input volume in the output as was discussed in detail during the course. Furthermore, in order to enhance the efficiency in allocating various parameter settings, hyperparameters are tuned in a "sweep approach". The extent in which it will operate is determined by the input values of which the process results will be displayed and documented throughout the report.

3 Convolution Neural Network

3.1 Implementation

The project entailed the development of an convolution neural network architecture for facial expression recognition. Therefore, the implementation was built upon the library TensorFlow as it has excellent support to run with the traditional Google engine of Colaboratory. The latter IDE was chosen over Pycharm as the notebook provides a connection with a single NVIDIA GPU to perform parallel computations with CUDA. This is necessary as some basic Tensorflow operations benefit from the provided speedup capability, especially when operating in a deep learning environment.

The results from different runs where tracked using Weights and Biases, a tool for experiment tracking. Herewith, hyperparameters are defined to tune upfront and run the experiments on multiple machines simultaneously. The upshift of this approach was a reduced coordination and experiment tracking overhead while everything was assembled in comprehensive graphs. On the other hand, a well-defined configuration with the tool was needed so that the data being tracked had a clear documentation.

3.2 Architecture

Firstly, a neural network was created with three blocks of convolutional layers, followed by a max pooling layer. Because of the insufficient performance of the model, the number of layers were increased. This resulted in a very similar structure of the neural network, which acquired incentive from the repository contributed by NJNischal[1]. It was used as a building block to appropriately enhance the influence of parameters during the experiments without the bias of constantly changing the inherent structure of the network.

More precisely, the network entails four convolution layers with at the end two fully connected blocks to form the final output in regard to the seven classes of emotion. After each convolutional and fully connected layer, batch normalization is firstly connected to accelerate the learning while the rectifier function is implemented to induce non-linear and interaction effects. Furthermore, to account for dimensionality reduction and normalization, Max-pooling and different dropout ratios are considered. In the end, a dense layer is extended with the Softmax activation function to assign the generated probabilities to each class in the emotion identifying problem.

3.3 Hyperparameter tuning

There are multiple ways to find the best hyperparameter, in the experiments three different types of searching are examined to find the best hyperparameters: grid, random and bayesian search. The grid search was performed at first, which tries every possible combination. While this method searches the feature space exhaustively it also requires a lot of experiments. To reduce the amount experiments a random search was applied, which combines the hyperparameters in a random manner. This has the disadvantage, that the random selection could never find the optimal combination. The final approach, bayesian search, in which a probability model of the objective function is build, based on this model the most promising hyperparameters are selected. A comparison of random and bayesian search is shown in figure 1. As is shown, bayes quickly finds the optimal parameters. In this case random is already pretty close to the optimal values in

[1] `https://github.com/NJNischal/Facial-Expression-Recognition-with-CNNs`

the beginning. This is most likely because the search space defined was already quite narrow.

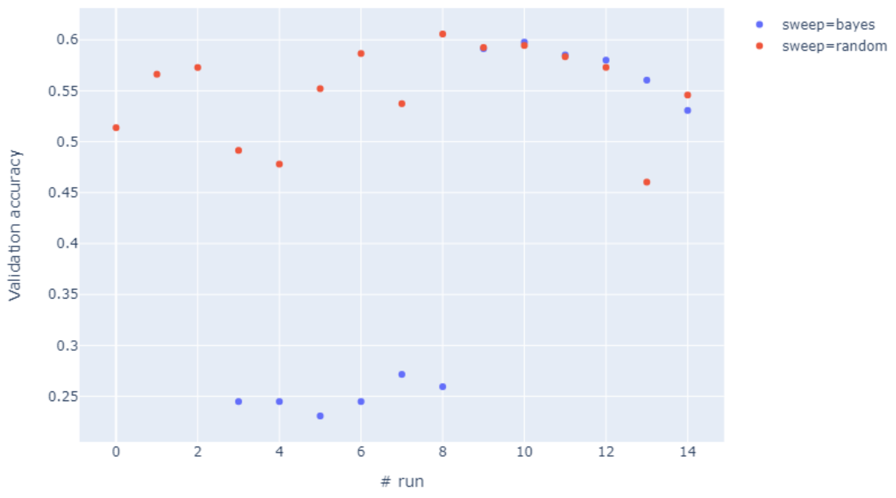


Figure 1: Random vs bayesian search

3.4 Experiments

3.4.1 Dimensionality output space

This section implies different combinations of dimensionalities over the convolution layers. It is now observed which combination or dimensionality chain amongst the four layers is applicable to entail a high accuracy. From figure 2, an initialization dimensionality of 64 or 128 in layer 1 captures essential properties to obtain a high accuracy. The highest accuracy of 61,7% is achieved by the set 64, 128, 1028, 512 in connection with the four layers. Moreover, another set 64, 256, 256, 512 reaches the second highest accuracy. Thus, there isn't on predefined value set of layers that could entail in an optimal solution. Therefore, arbitrary dimensionalities of the filters have to be evaluated continuously in order to find that optimal combination with respect to the other adjusted hyperparameters.

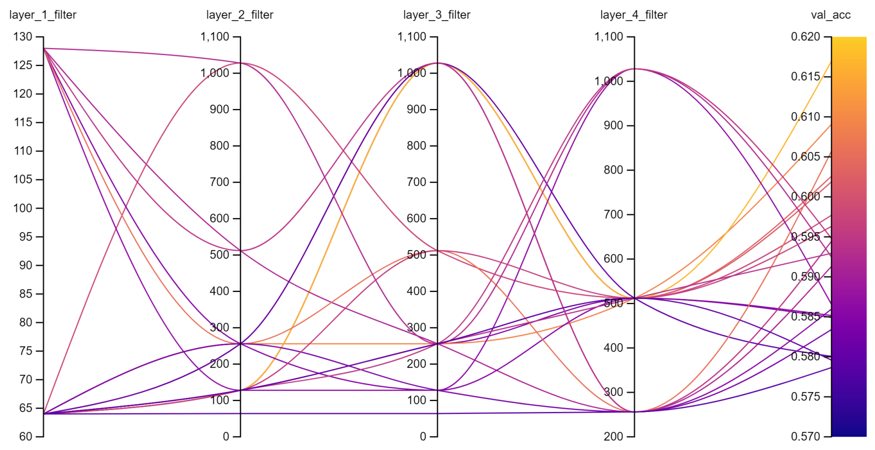


Figure 2

3.4.2 Dropout

At each training stage, dropout is considered to prevent over-fitting of the model as it drops a fraction of the input units. From the graph illustrated in figure 3, a dropout with probability equal to 25% will drop this fraction while retaining the majority of the input values. From the conducted experiments, it is clear that the same fraction of 25% will result in an optimal setting. Notice that a dropout of 50% has a counterproductive effect when compared to no dropout or dimensionality reduction.

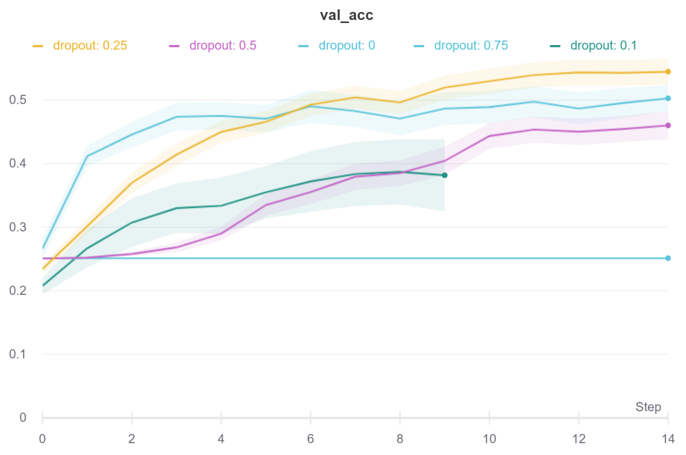


Figure 3

3.4.3 Learning rate

In the introduction, it was shortly mentioned that the learning rate was configured at 0.0005 as it shows reasonable performance in figure 4. Although, it is one of the most important hyperparameters, most of the experiments remained being executed with this rate. But, an additional approach was implemented to interactively change the learning rate. When a plateau in model performance is detected, a callback will reduce the learning rate with the aim to fine-tune model weights and induce a further improvement in validation accuracy. Thus, when the accuracy has stopped improving after a certain length of epochs, the learning rate is reduced as in figure 5.

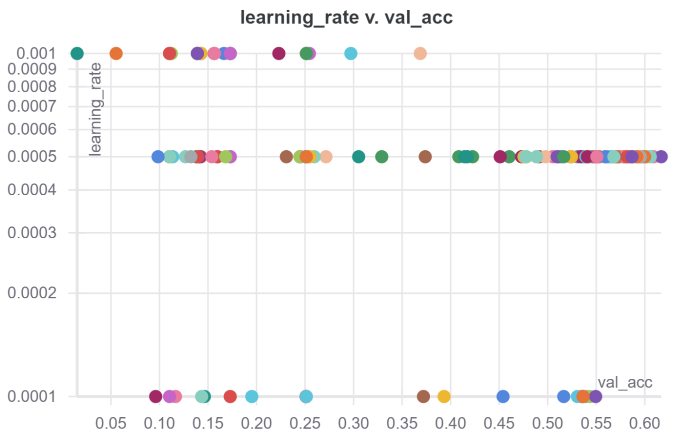


Figure 4

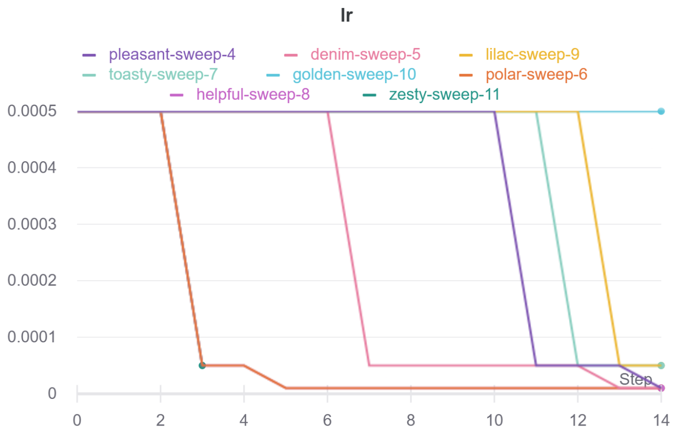


Figure 5

3.4.4 Loss function

In general, cross-entropy and mean squared error are the two types of loss functions being used in training neural networks to evaluate a set of weights. As the weights are being adapted to enhance a minimized loss, it is of utmost importance that the errors in testing are reflected by the right loss function technique. In the case of a classification problem, the probability of belonging to each of the seven classes needs to be predicted. Cross-entropy is used in most of the experiments to estimate the relative entropy between two probability distributions. Moreover, cross-entropy showed to be a better fit over mean squared error as the latter doesn't punish misclassifications enough. Thus, it is more suited for regression than the classification of labels. Therefore, only the technique of cross-entropy is displayed in figure 6.

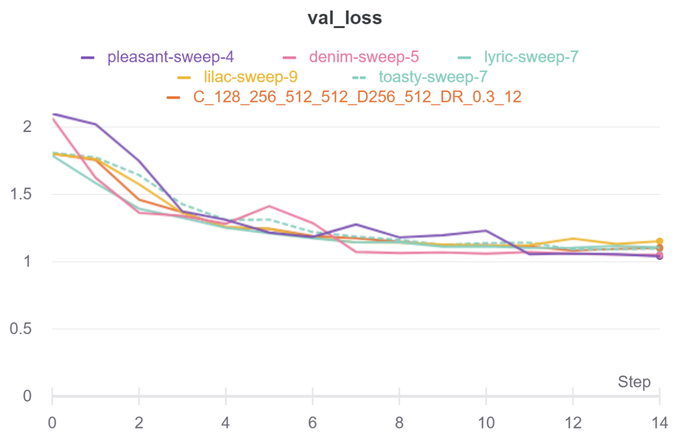
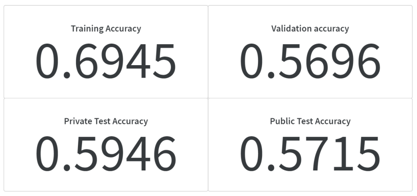


Figure 6

3.5 Performance

To validate the performance of the model, parts of the FER-2013 challenge were included with the label "private test" and "public test". Notice that both data sets weren't used during training. In figure 7, the accuracy on the different data sets is displayed.



Training Accuracy	Validation accuracy
0.6945	0.5696
Private Test Accuracy	Public Test Accuracy
0.5946	0.5715

Figure 7

In the following analysis, the gradients of the model are analyzed. The gradients show how the models weights changed during training:

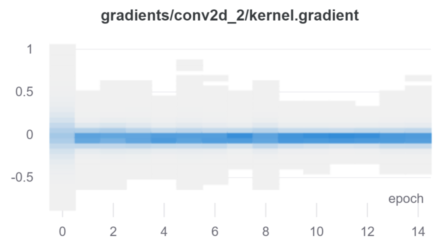


Figure 8

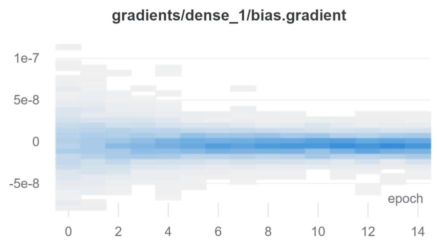


Figure 9

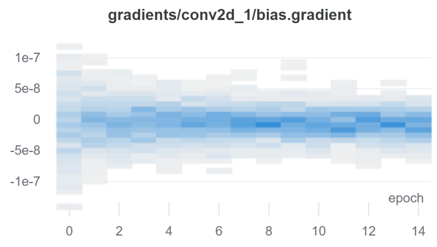


Figure 10

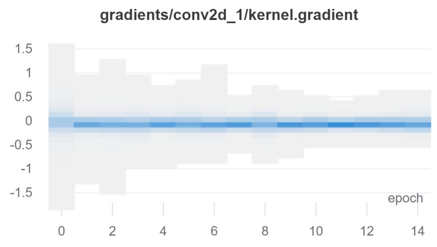


Figure 11

Here predictions for different emotions that the model predicted are displayed as an illustration:

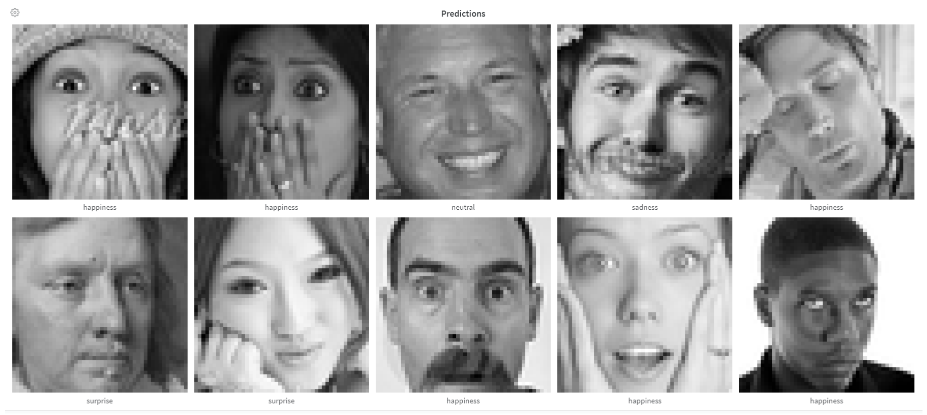


Figure 12

As can be seen in the confusion matrix, the model often misclassified images as ”neutral” or ”sadness”. These are the areas where the model could improve the most.

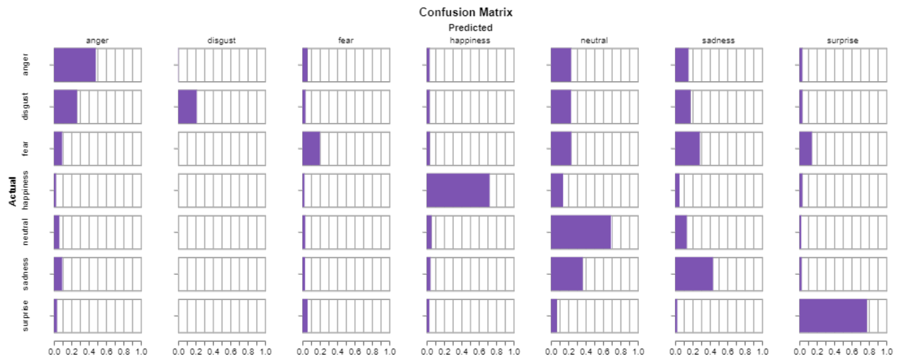


Figure 13

4 Conclusion

An automatic method for emotion identification was presented in this report. Once it was optimized, it could classify the contents of different images to an associated emotion. Throughout the experiments, it was important to remember the use case and its corresponding image data. As a consequence, the input shape of images induces some important limitations on the structure of the neural network. It was quickly analysed that four layers with kernel size (3x3) were the right fit for convolving and extracting relevant features. Furthermore, the evaluation of the model was focused on the observation of training and especially the validation accuracy as it presented our control metric. As the comparison between these two had a small gap in figure 7, it indicates a slight beginning of over-fitting in our model. We prevented this as much as possible by a correct split of data, shuffling to a balanced label batch, a learning rate reduction on plateau and dropout. As showed before the model improved the most, when dropout was introduced. While we can say that the trained model is able to classify emotions, we notice that neutral or sad emotions pose a challenge for the current model. Our exhaustive search suggest, that the model provided is not complex enough to learn more insights.