IN SUPREMÆ DIGNITATIS 1343

UNIVERSITÀ DI PISA

Computer Engineering

Intelligent Systems Project

Project Documentation

Gaia Anastasi

Academic Year 2020/2021

Contents

1 Introduction

The project is composed by 2 parts. The aim of the fist part is to design and develop an intelligent system which measures a person's affective state using signals recorded by sensors while the aim of the second part is to develop a classifier of emotions using face images as inputs.
The task performed in this project are:

- **3.1:** Design and development of two MLP and two RBF that accurately estimate a person's valence and arousal levels, respectively.
- **3.3:** Design and development of a fuzzy inference system to fix the deficiencies in the arousal dimension.
- **4.2:** Fine-tuning of a pre-trained CNN to accurately classify a person's emotions based on facial expressions.

2 Design and develop of an intelligent system

2.1 Data Balancing

Before implementing the first task of the project, a pre-processing of the dataset was needed. The steps done are:

- Removal of non-numerical or infinite values using **MATLAB** function *isinf* to verify the presence of such values
- Removal of outliers using **MATLAB** function *rmoutliers* and the default *median* method.
- Balacing of the dataset
- Features selection
- Normalization of data

The last 3 items will be discussed in the following sections.

2.1.1 Balancing of tha dataset

The dataset is composed by 54 signals and for each of them a value for valence and one for arousal are presented. Considering that valence and arousal can take 7 different values, the dataset can be split in 7 classes. The distribution of samples among the classes is the following:

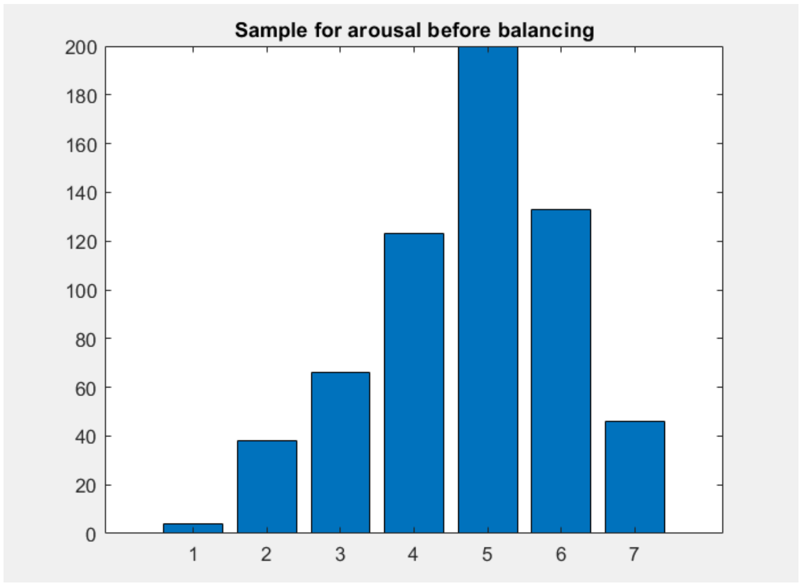


Figure 1: Samples distributions for arousal before balancing

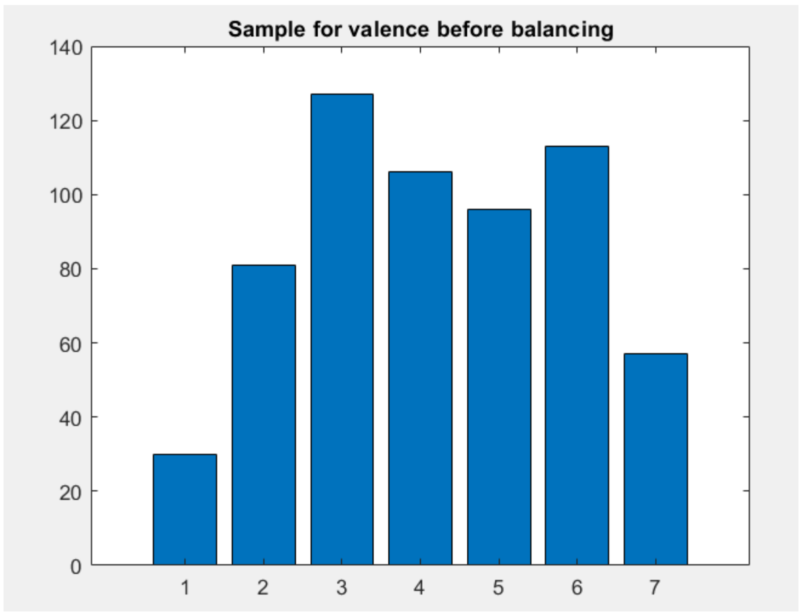


Figure 2: Samples distributions for valence before balancing

The samples are unbalanced so the following algorithm based on undersampling, oversampling and data augmentation was used to balance the dataset. Data augmentation is applied on samples which belong to the least common class for arousal(valence) and not to the most common class for valence(arousal) and then the samples which belong to the most common class for arousal(valence) and don't belong to the least common class for valence(arousal) are removed. These steps were repeated *rep* times.
After performing the balancing, these are the distributions of samples for both

the arousal and the valence:

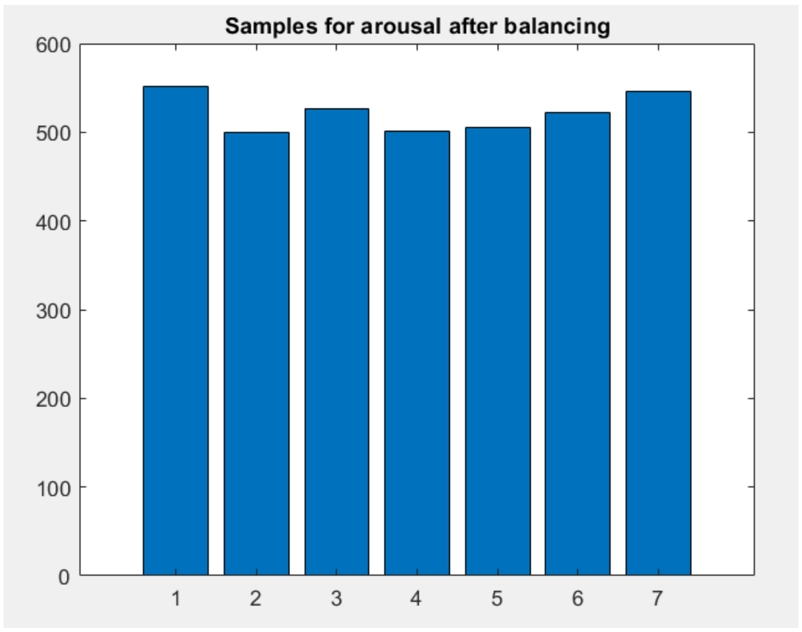


Figure 3: Samples distributions for arousal after balancing

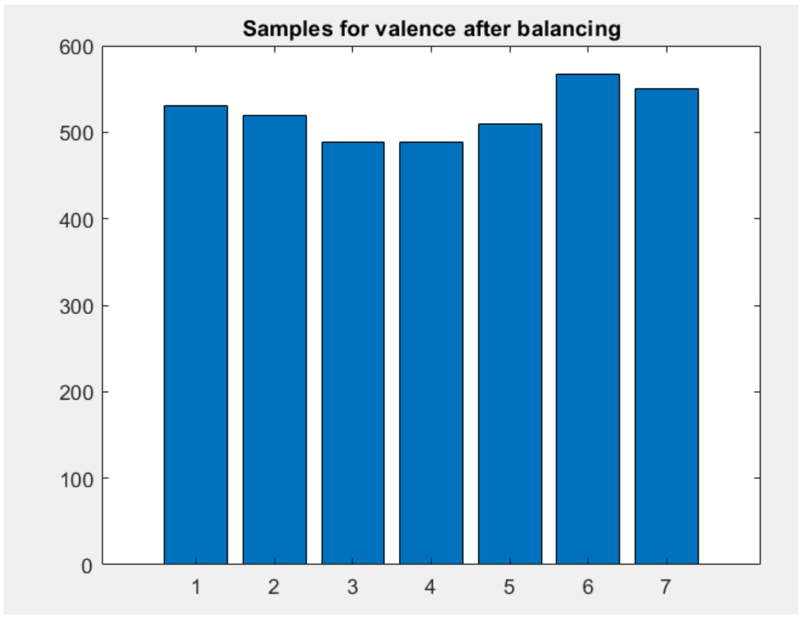


Figure 4: Samples distributions for valence after balancing

2.1.2 Feature selection

To select the features **MATLAB** function *sequentialfs* was used. The function should be repeated a statistically relevant number of times to extract the right features, but in order to perform the algorithm in an acceptable amount of time in this project the *sequentialfs* function was repeated just 5 times. At the end of the algorithm 5 **.mat** files were generated:

- *training-arousal.mat*: which contains the training inputs and corresponding target outputs for arousal training
- *testing-arousal.mat* which contains the testing inputs and corresponding target outputs for arousal testing
- *training-valence.mat* which contains the training inputs and corresponding target outputs for valence training
- *testing-valence.mat* which contains the testing inputs and corresponding target outputs for valence testing
- *best3.mat* which is a structure containing the training data and testing data for arousal of the best 3 features selected. This dataset will be used for the task 3.3

2.1.3 Normalization of data

The data were normalized using the **MATLAB** function *normalize* which takes as an argument the data that need to be normalized. For all the other parameters the default values were used.

2.2 Multi-Layer Perceptron Networks

In this chapter the design and development of 2 MLP networks to accurately estimate a person's valence and arousal levels, respectively, will be discussed. The experiments reported here were initially performed on data not normalized, but the same experiments were performed as well on normalized data. The results of the latter ones were not reported here because no significant changes were found between the two cases. Some experiments were performed in order to determine which MLP architecture is the best for the project purpose.

2.2.1 Arousal MLP Network

Network configurations:

```
hiddenLayerSize_arousal = 30;
mlp_arousal = fitnet(hiddenLayerSize_arousal);
mlp_arousal.divideParam.trainRatio = 0.7;
mlp_arousal.divideParam.testRatio = 0.1;
mlp_arousal.divideParam.valRatio = 0.2;
mlp_arousal.trainParam.epochs =110;
```

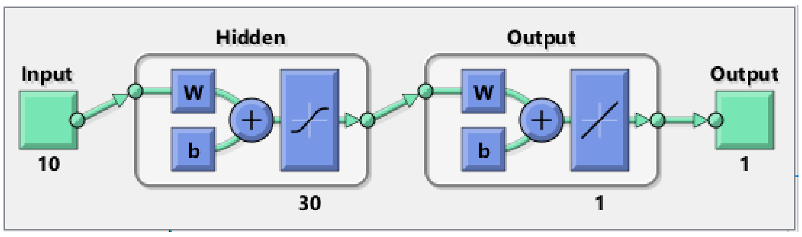


Figure 5: Arousal MLP network architecture

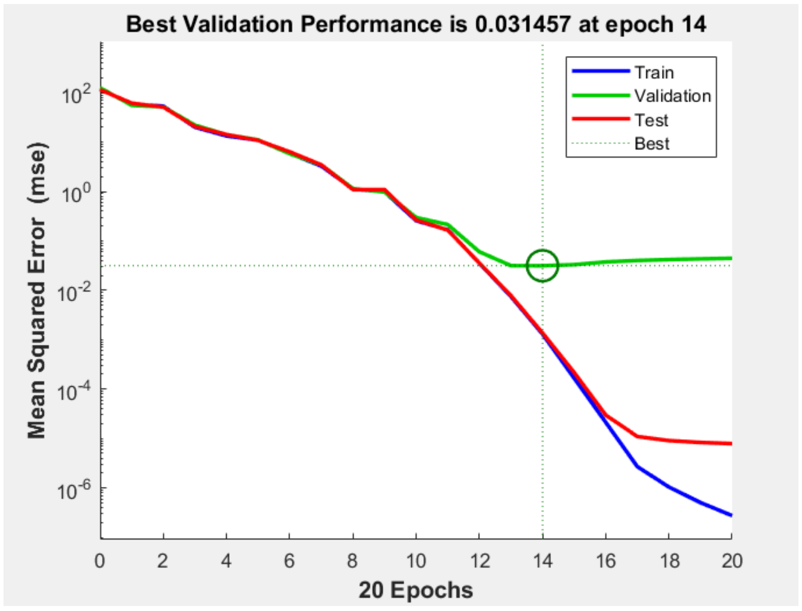


Figure 6: Performance for arousal MLP network

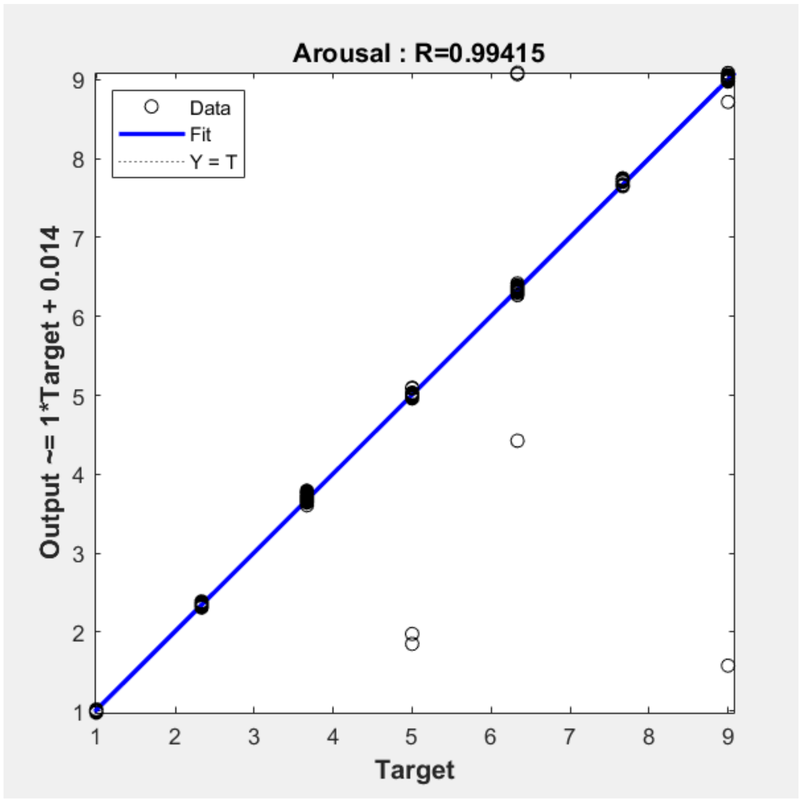


Figure 7: Regression for arousal MLP network

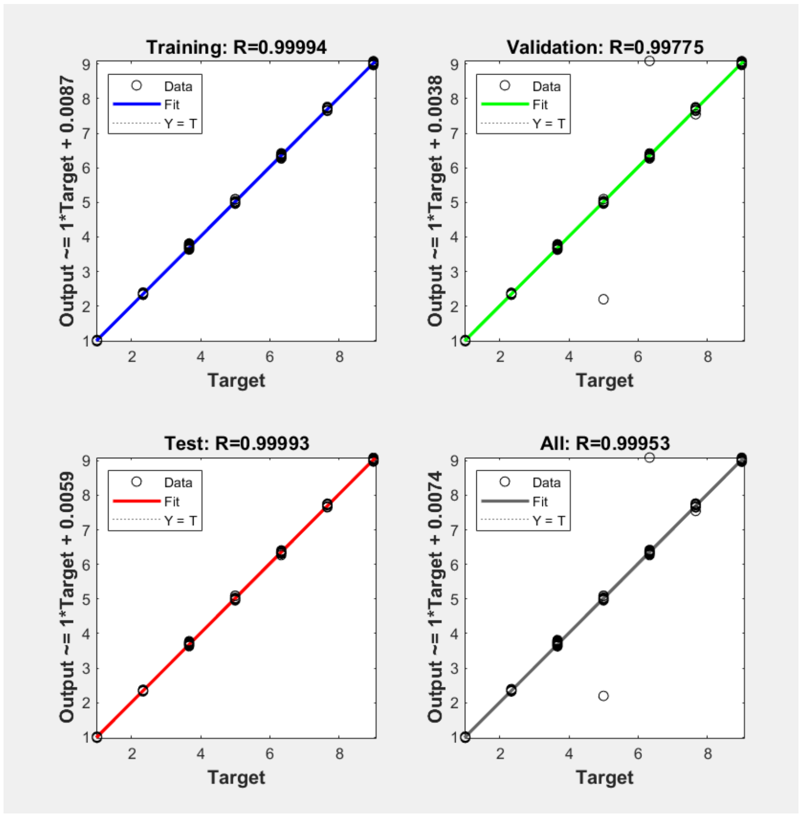


Figure 8: Regression results after training for arousal MLP network

2.2.2 Valence MLP Network

Network configuration:

```
hiddenLayerSize_valence = 35;
mlp_valence = fitnet(hiddenLayerSize_valence);
mlp_valence.divideParam.trainRatio = 0.7;
mlp_valence.divideParam.testRatio = 0.1;
mlp_valence.divideParam.valRatio = 0.2;
mlp_valence.trainParam.epochs =110;
```

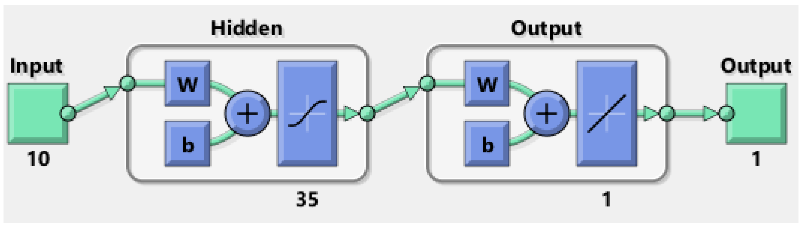


Figure 9: Valence MLP network architecture

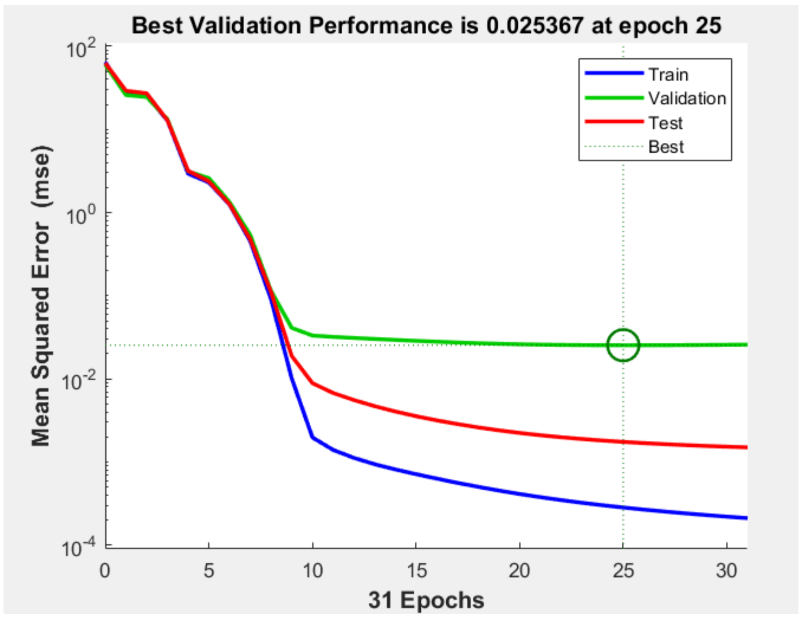


Figure 10: Performance for valence MLP network

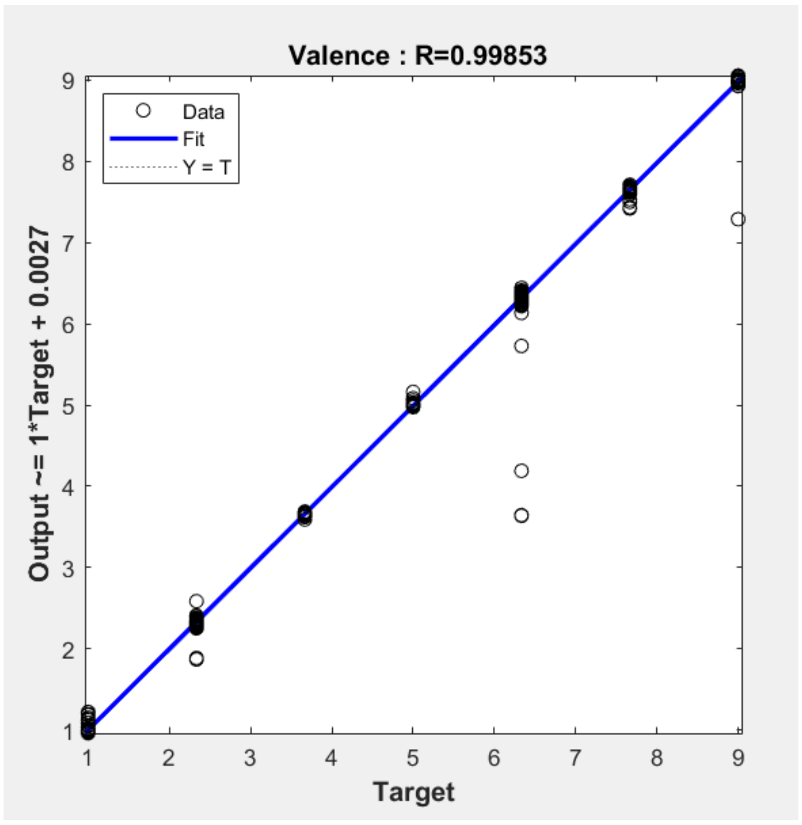


Figure 11: Regression for valence MLP network

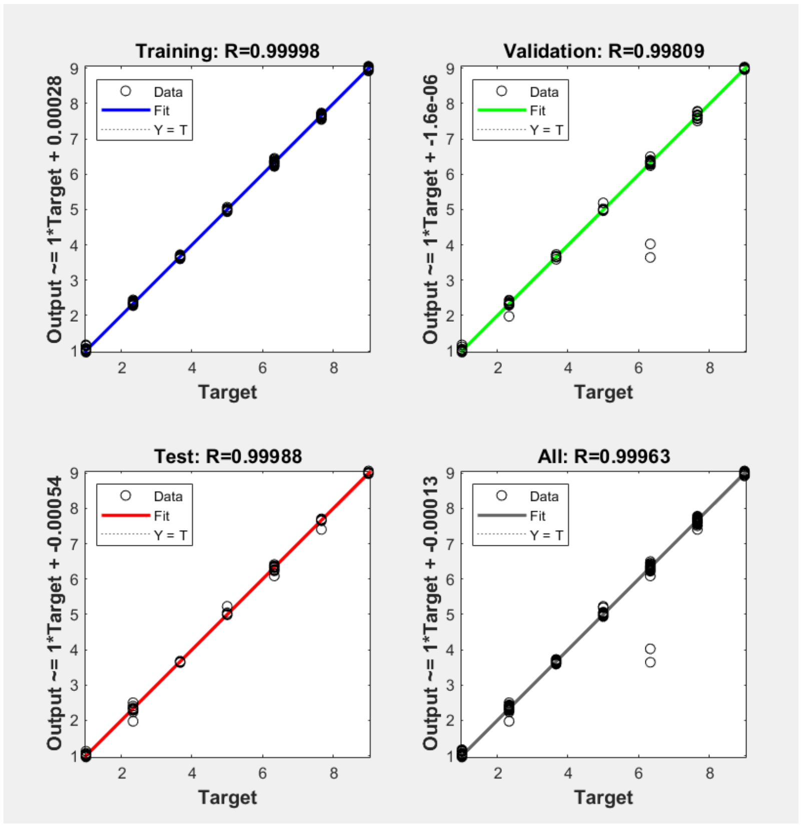


Figure 12: Regression results after training for valence MLP network

In general, valence MLP network requires more hidden neurons to perform as good as the arousal MLP network.

2.3 Radial Basis Function Network

In this chapter the design and development of 2 RBF networks will be discussed. Again the experiments were performed on both normalized and not normalized data, but just the results about not normalized ones were reported here. Some experiments were performed in order to determine which RBF architecture is the best for the project purpose.

2.3.1 Arousal RBF network

Network configurations:

```
goal_ar = 0;
spread_ar = 1;
K_ar = 400;
```

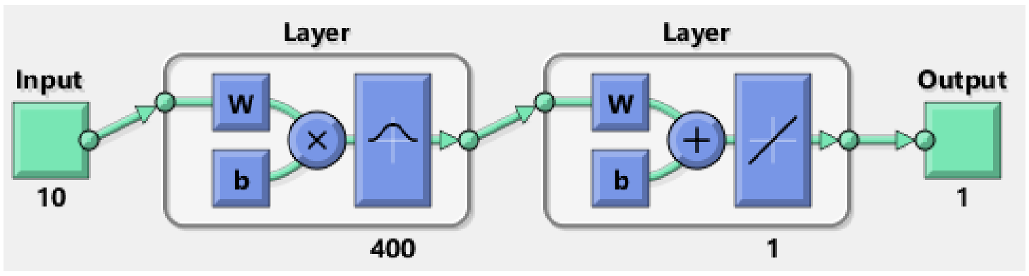


Figure 13: Arousal RBF network architecture

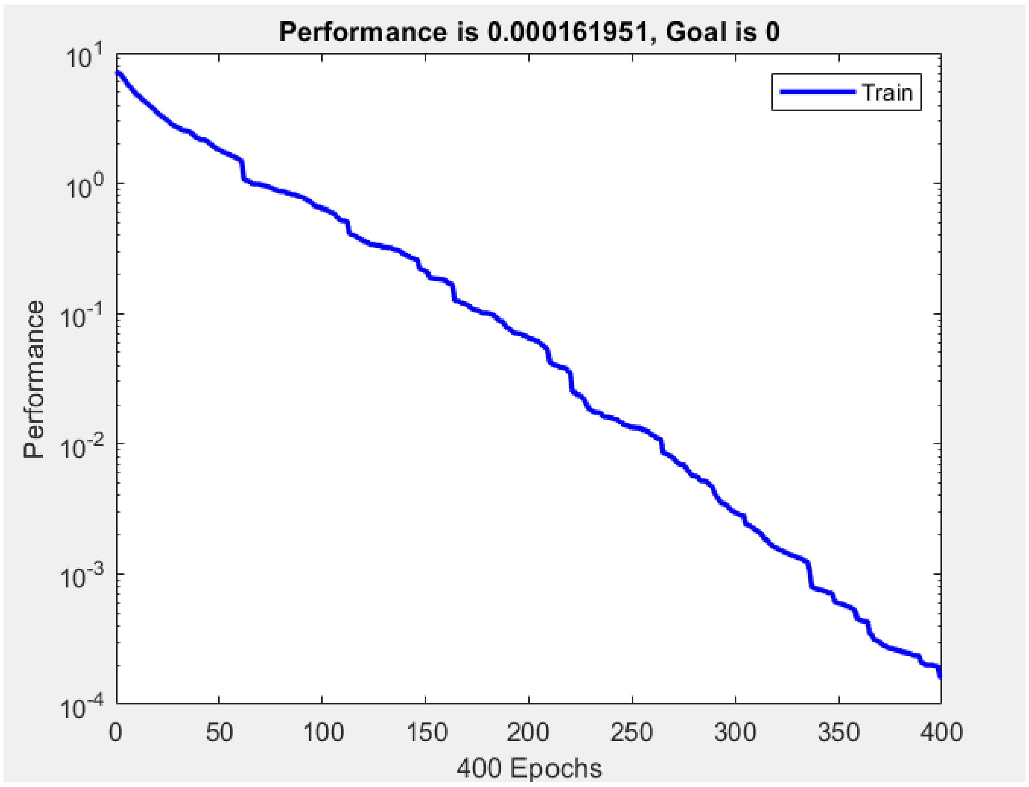


Figure 14: View arousal RBF network

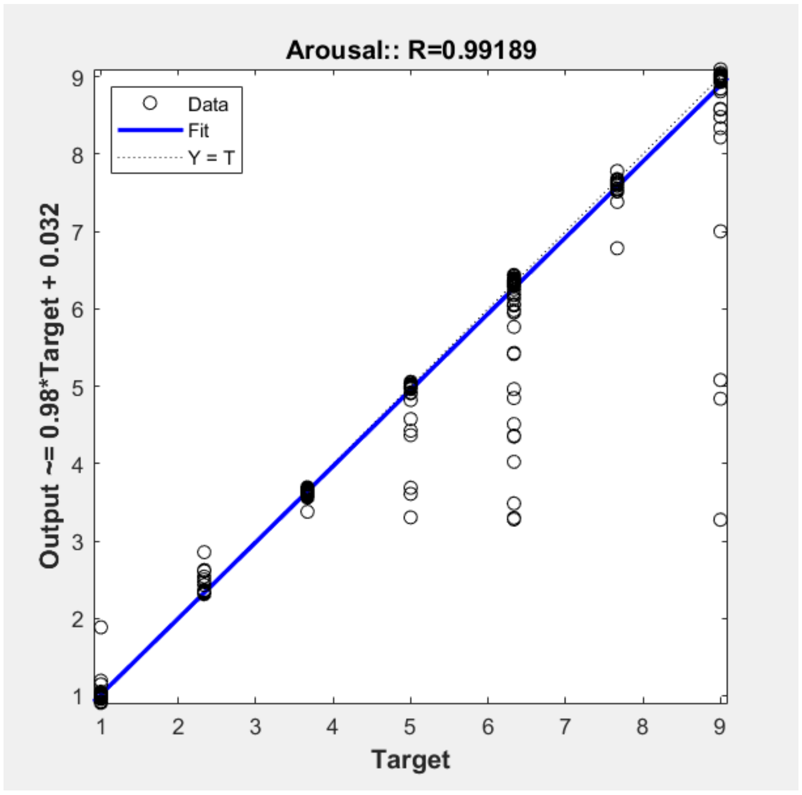


Figure 15: Regression for arousal RBF network

2.3.2 Valence RBF network

Network configurations:

```
goal_ar = 0;
spread_ar = 1;
K_ar = 350;
```

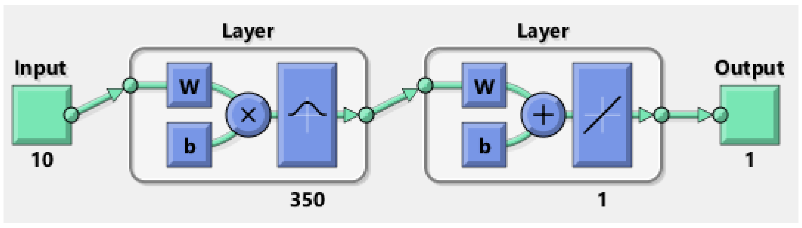


Figure 16: Valence RBF network architecture

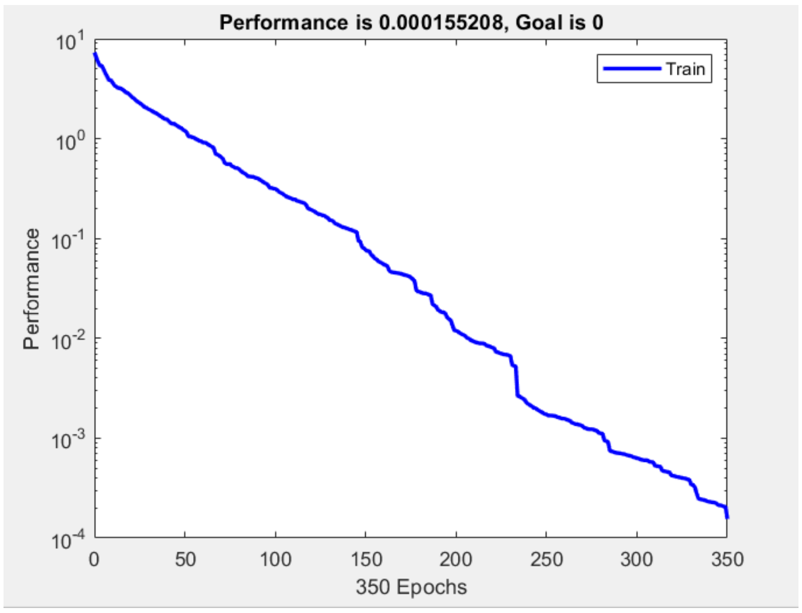


Figure 17: View valence RBF network

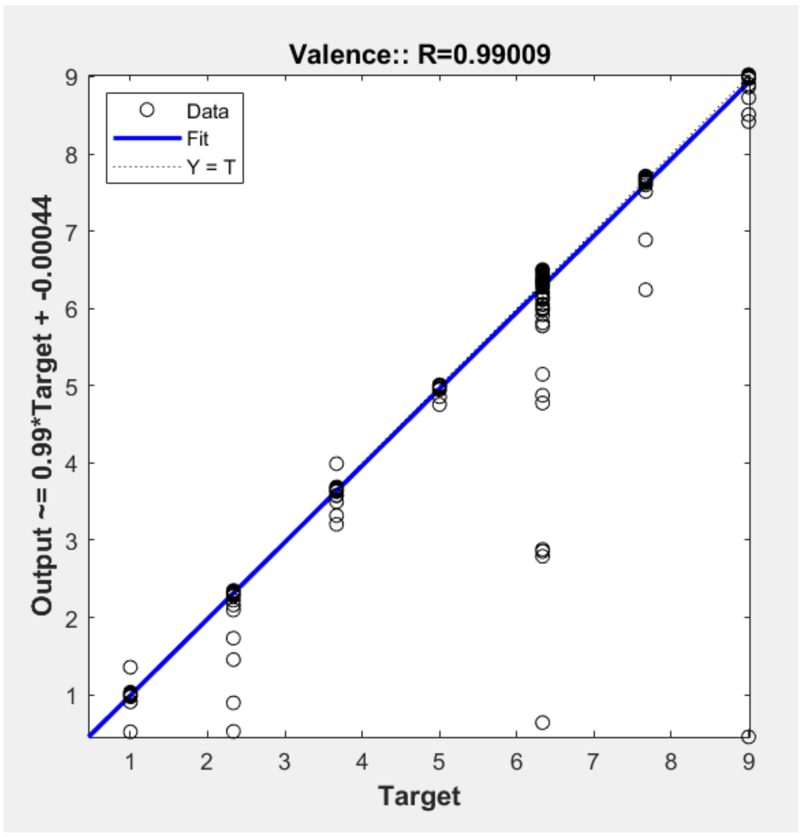


Figure 18: Regression for valence RBF network

In general, arousal RBF network requires more epochs to perform as good as the valence one.

2.4 Mamdani-type fuzzy inference system

The aim of this section is to show how a fuzzy inference system which fixes deficiencies in the arousal dimension was designed and developed. At the beginning the **empical distribution** of the features through some histograms was done.

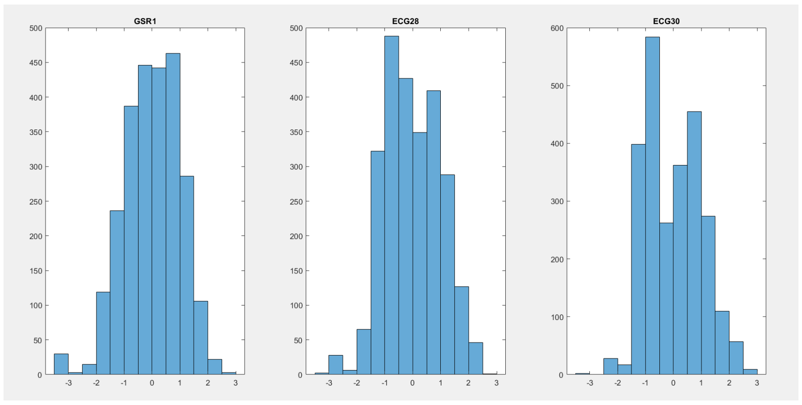


Figure 19: Empirical distribution of the features

These distributions have been used to model the linguistic variables of the three features. Since these distributions tend to have some peaks these peaks were considered as a central point of a particular linguistic label.

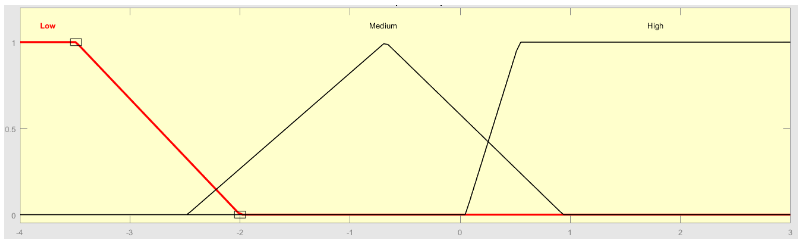


Figure 20: Modeling for linguistic variable of feature 27: GSR1

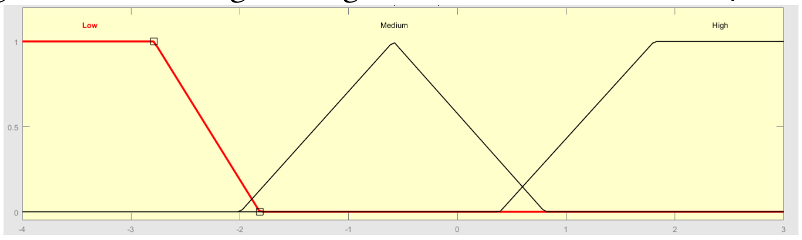


Figure 21: Modeling for linguistic variable of feature 11: ECG28

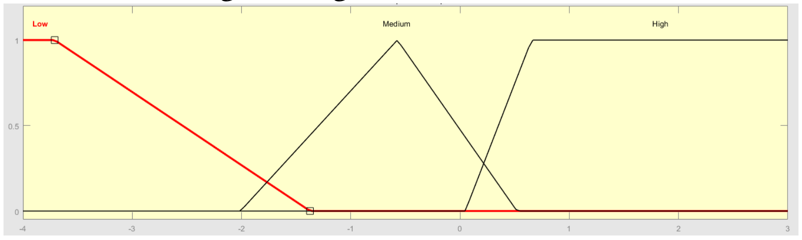


Figure 22: Modeling for linguistic variable of feature 13: ECG30

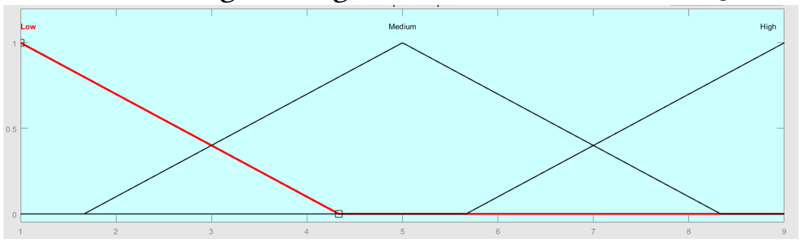


Figure 23: Modeling for linguistic variable of the output

For what concern the rules, other analysis have been performed. In particular a **correlation analysis** has been performed using different combination of the input features to check possible correlations among the selected features. As we can see from the graphs below there is correlation among the features. Plus, other histograms has been plotted considering this time just a subset of the samples coinciding to a specific range of arousal. As we have seen during the pre-processing the possible values of arousal can be divided in 7 classes, so the first and the second class were used to implement the low range of arousal, third, forth and fifth classes to implement the medium range and final sixth and seventh to implement the high range.

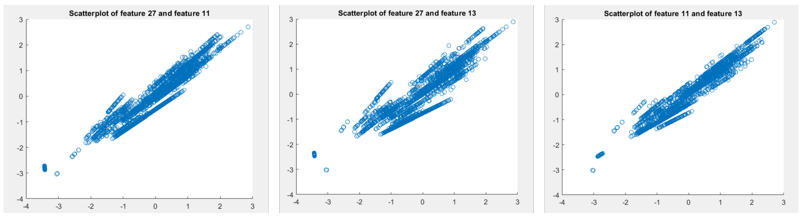


Figure 24: Correlation study

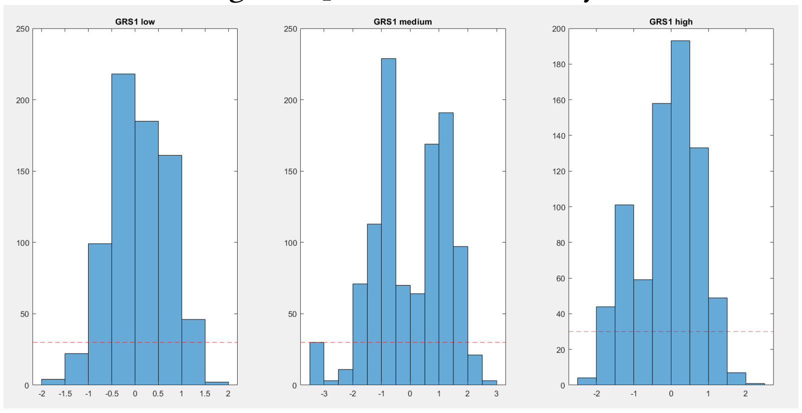


Figure 25: Histogram of feature GSR1(27) with different range of arousal

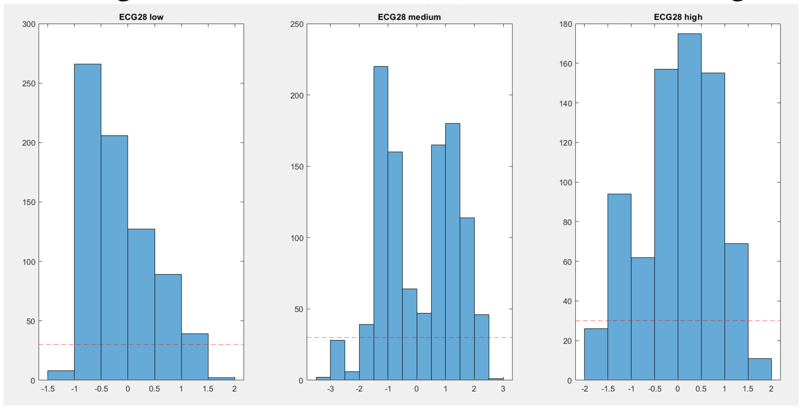


Figure 26: Histogram of feature ECG28(11) with different range of arousal

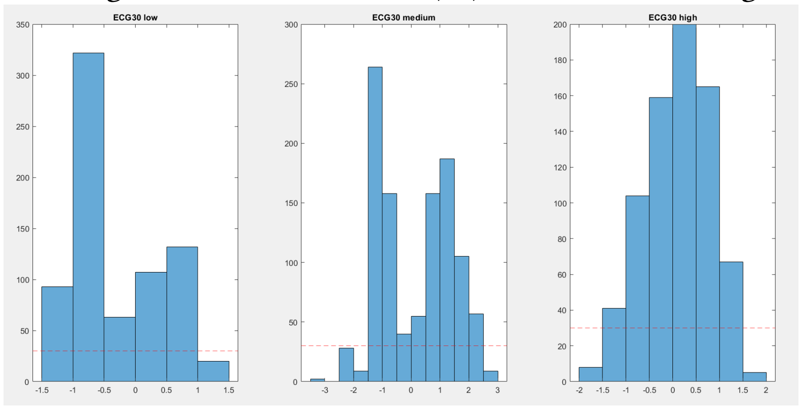
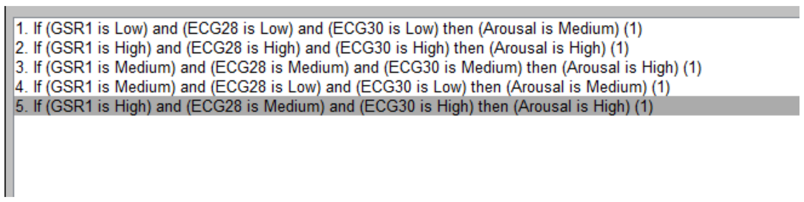


Figure 27: Histogram of feature ECG30(13) with different range of arousal

The rules modelled are the following:



```
1. If (GSR1 is Low) and (ECG28 is Low) and (ECG30 is Low) then (Arousal is Medium) (1)
2. If (GSR1 is High) and (ECG28 is High) and (ECG30 is High) then (Arousal is High) (1)
3. If (GSR1 is Medium) and (ECG28 is Medium) and (ECG30 is Medium) then (Arousal is High) (1)
4. If (GSR1 is Medium) and (ECG28 is Low) and (ECG30 is Low) then (Arousal is Medium) (1)
5. If (GSR1 is High) and (ECG28 is Medium) and (ECG30 is High) then (Arousal is High) (1)
```

Figure 28: Fuzzy rules

3 Classification of emotions using facial expression

In the following chapter the development and the training of a CNN based on the pre-trained AlexNet will be discussed. The network will be used to classify facial expressions. The possible classes are:

- Anger
- Disgust
- Fear
- Happiness

3.0.1 Image selection

Since the dataset provided is unbalanced, an image selection was performed. For each class 500 images were selected randomly at the beginning, in the second experiment the number of images for each classes was brought to 1000. Finally, a new experiment was performed using 500 images for each class and each image was accurately selected among the ones provided. For both experiments 70% of these images will be used for training, 20% for validation and 10% for testing.

3.0.2 Design Choices

All the images used were resized in order to fit the input size required by AlexNet. Moreover, a random translation of 30 pixels in horizontally and vertically way was performed as well on the training set in order to improve performances and prevent overfitting.

```
pixelRange = [-30 30];
imageAugmenter = imageDataAugmenter( ...
    'RandXReflection',true, ...
    'RandXTranslation',pixelRange, ...
    'RandYTranslation',pixelRange);

augmented_image_data_train = augmentedImageDatastore(
input_size(1:2), data_train,
'DataAugmentation', imageAugmenter);
augmented_image_data_validation = augmentedImageDatastore(
input_size(1:2), data_validation);
augmented_image_data_test = augmentedImageDatastore(
input_size(1:2), data_test);
```

The last 3 layers of AlexNet were substituted with other layers. This was made because AlexNet is trained to classify thousands of categories, but in this project there is just 4 classes in which data need to be classified.

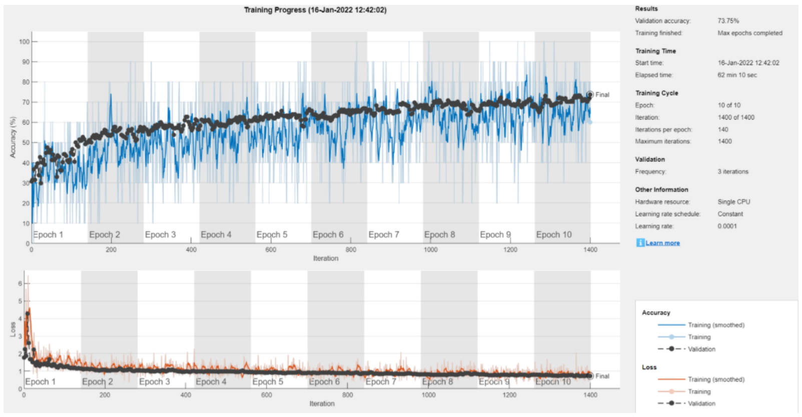
```
layers = [
    original_layers
    fullyConnectedLayer(numClasses, 'WeightLearnRateFactor',20,
    'BiasLearnRateFactor',20)
    softmaxLayer
    classificationLayer];
```

The following hyper-parameters were used to train the network:

```
options = trainingOptions('sgdm', ...
    'MiniBatchSize', 10, ...
    'MaxEpochs', 10, ...
    'InitialLearnRate', 1e-4, ...
    'Shuffle', 'every-epoch', ...
    'ValidationData',augmented_image_data_validation, ...
    'ValidationFrequency', 3, ...
    'Verbose', false, ...
    'Plots', 'training-progress')
```

3.0.3 Final Results

In this section the final results about the CNN will be reported. In the case of 500 images randomly chosen for each classes, the network has to perform less iteration before ending, but as we can clearly see from the confusion matrix, performances aren't that great. In particular the network has some difficulty in recognising anger facial expressions and distinguishing them from disgust ones. The only facial expression that is detected with acceptable percentage is actually happiness that is the most different from the other ones and for this reasons also the most easily recognisable.



Results

Validation accuracy:	73.75%
Training finished:	Max epochs completed

Training Time

Start time:	16-Jan-2022 12:42:02
Elapsed time:	62 min 10 sec

Training Cycle

Epoch:	10 of 10
Iteration:	1400 of 1400
Iterations per epoch:	140
Maximum iterations:	1400

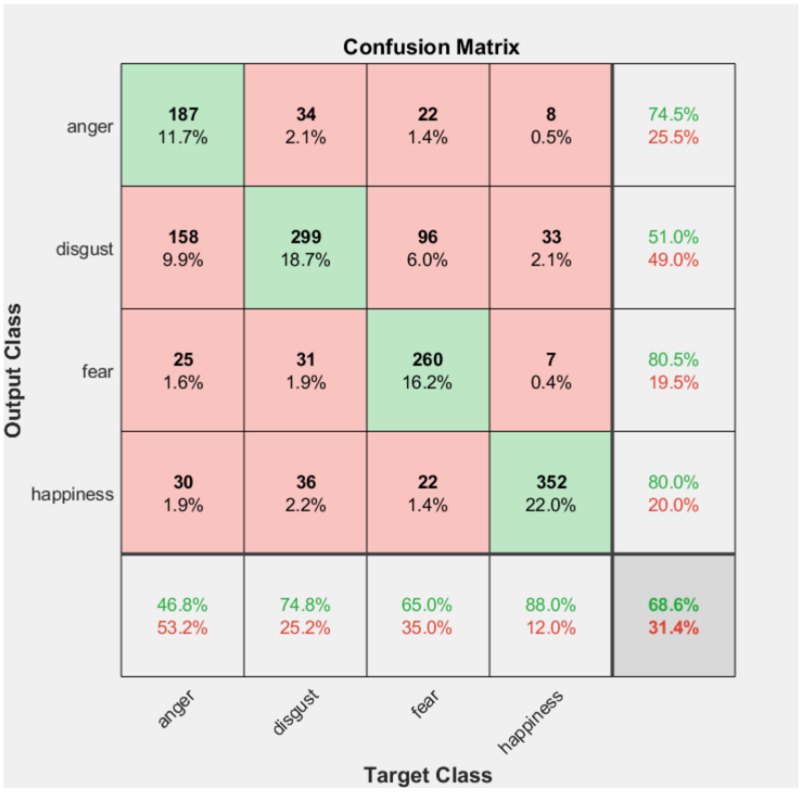
Validation

Frequency:	3 iterations

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	0.0001

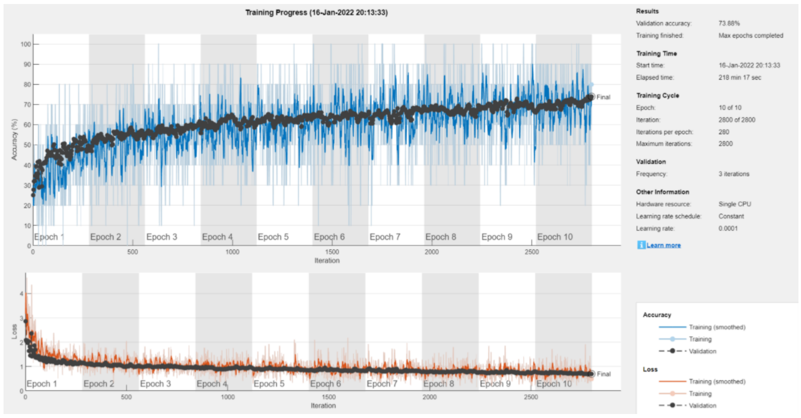
Figure 29: Training process of a CNN with 4 classes and 500 images for each class as inputs


Confusion Matrix
Output Class
Target Class

	anger	disgust	fear	happiness	
anger	187 11.7%	34 2.1%	22 1.4%	8 0.5%	74.5% 25.5%
disgust	158 9.9%	299 18.7%	96 6.0%	33 2.1%	51.0% 49.0%
fear	25 1.6%	31 1.9%	260 16.2%	7 0.4%	80.5% 19.5%
happiness	30 1.9%	36 2.2%	22 1.4%	352 22.0%	80.0% 20.0%
	46.8% 53.2%	74.8% 25.2%	65.0% 35.0%	88.0% 12.0%	**68.6% 31.4%**

Figure 30: Confusion matrix of a CNN with 4 classes and 500 images for each class as inputs

Increasing the number of images used for each classes we can notice that performances improve. This is due to the fact that we have more data to train our network on. As we can see in the following pictures, especially in the confusion matrix, the TPR in detecting correctly anger and fear have increased. Happiness remains the easiest recognisable facial expression among the considered ones.



Results

Validation accuracy:	73.88%
Training finished:	Max epochs completed

Training Time

Start time:	16-Jan-2022 20:13:33
Elapsed time:	218 min 17 sec

Training Cycle

Epoch:	10 of 10
Iteration:	2800 of 2800
Iterations per epoch:	280
Maximum iterations:	2800

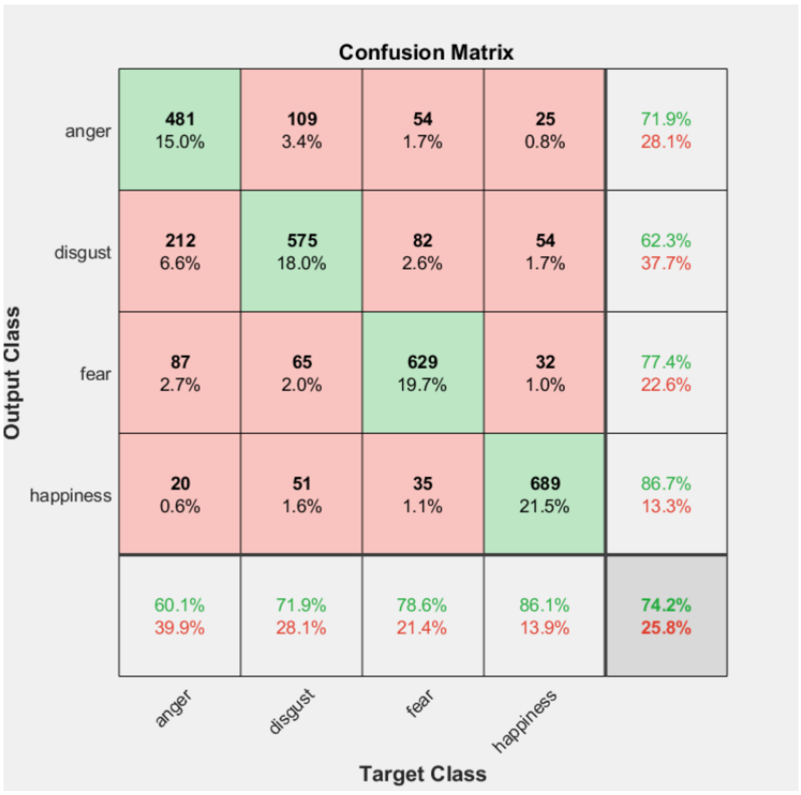
Validation

Frequency:	3 iterations

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	0.0001

Figure 31: Training process of a CNN with 4 classes and 1000 images for each class as inputs

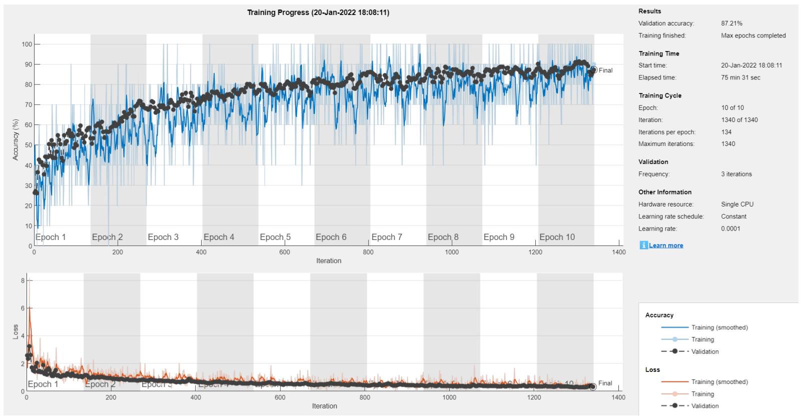


Confusion Matrix

Output Class \ Target Class	anger	disgust	fear	happiness	
anger	481 15.0%	109 3.4%	54 1.7%	25 0.8%	71.9% 28.1%
disgust	212 6.6%	575 18.0%	82 2.6%	54 1.7%	62.3% 37.7%
fear	87 2.7%	65 2.0%	629 19.7%	32 1.0%	77.4% 22.6%
happiness	20 0.6%	51 1.6%	35 1.1%	689 21.5%	86.7% 13.3%
	60.1% 39.9%	71.9% 28.1%	78.6% 21.4%	86.1% 13.9%	74.2% 25.8%

Figure 32: Confusion matrix of a CNN with 4 classes and 1000 images for each class as inputs

The last experiment was performed choosing accurately 500 images as inputs for each class. During the selection the images which have ambiguous facial expressions were removed. If we compare this experiment with the one where we have chosen 500 images randomly we can notice that the validation accuracy increases from 73.75% to 87.21% and TPR increases as well while FPR decreases as we can clearly see from the confusion matrix below. Performance increases as well if we compare these results with the results obtain in the case of 1000 images picked randomly.



Results

Validation accuracy:	87.21%
Training finished:	Max epochs completed

Training Time

Start time:	20-Jan-2022 18:08:11
Elapsed time:	75 min 31 sec

Training Cycle

Epoch:	10 of 10
Iteration:	1340 of 1340
Iterations per epoch:	134
Maximum iterations:	1340

Validation

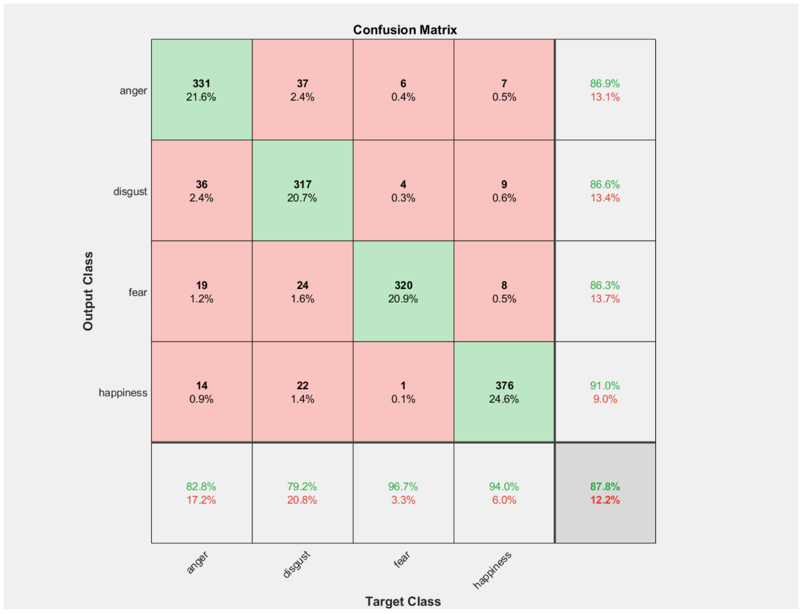
Frequency:	3 iterations

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	0.0001

Learn more

Figure 33: Training process of a CNN with 4 classes and 500 images accurately selected for each class as inputs



	anger	disgust	fear	happiness	
anger	331 21.6%	37 2.4%	6 0.4%	7 0.5%	86.9% 13.1%
disgust	36 2.4%	317 20.7%	4 0.3%	9 0.6%	86.6% 13.4%
fear	19 1.2%	24 1.6%	320 20.9%	8 0.5%	86.3% 13.7%
happiness	14 0.9%	22 1.4%	1 0.1%	376 24.6%	91.0% 9.0%
	82.8% 17.2%	79.2% 20.8%	96.7% 3.3%	94.0% 6.0%	87.8% 12.2%

Figure 34: Confusion matrix of a CNN with 4 classes and 500 images accurately selected for each class as inputs

As we can see performance improved a lot especially comparing the case in which we choose 500 images randomly for each class and the case in which we accurately select 500 images for each class. This is due to the fact that in the latter case the images provided are less ambiguous and this leads the network to make fewer mistakes.