Rock Paper Scissor

Alessandro Capici

July 7, 2021

1 The data

1.1 Introduction

Rock Paper Scissors is a two-players game in which each player simultaneously forms one of three shapes with an outstretched hand.
A data set has been collected by taking 2188 samples of the three forms. It is divided into a training (1888), a validation (150) and a test (150) set.

1.2 Samples

Figure 1: Example of images.

Taking a look at the samples we can see that all the images represent a hand in all 3 forms (rock, paper, scissors) resting on a green surface. All the images were captured always following the same hand position, all hands are oriented horizontally with respect to the x axis of the photo and in all photos the fingers are oriented to the left as we can see in the Figure 1.
This could be a good indicator on the use of low level features of the type of edge direction, because in this particular case we do not run into problems of rotation and translation, however some images are not perfectly centered with the table frame and this could cause some problems. As for all those low-lever features concerning colors, it is difficult to imagine that their use is fundamental to discriminate the various shapes since many images have different degrees of illumination and in addition there are only 2 colors, the green background and the hand color. On the other hand, the color low level features can be used to find a difference in green depending on the shape of the hand, there will be more green for rock, a little less for scissors and even less for paper.

2 Preprocessing

To make an image suitable to be processed by any model, a preprocessing procedure must be carried out, we are going to see 2 types.

2.1 Low Level Feature

- **Color low level features** divided in :
  - **color Histogram**: Gray-level images of shape (m, n) are first converted to RGB.
  - **RGB co-occurrence matrix**: The matrix represents the distribution of values of neighbor pixels. Color images are first converted to grayscale.
- **Direction low level features** divided in :

– **Edge direction histogram**: is computed by grouping the edge pixels which fall into edge directions and counting the number of pixels in each direction.

– **Co-occurrence matrix** : The matrix represents the distribution of values of neighbor pixels. Color images are first converted to grayscale.

2.2 PVMLNet

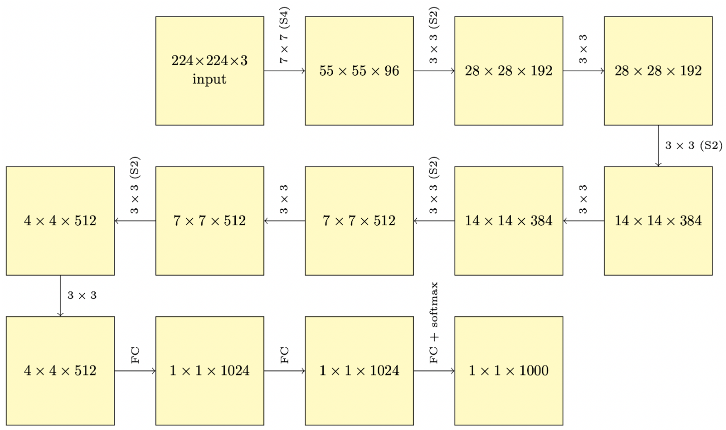

Figure 2: A sequence of eight "same size" convolutions followed by three fully-connected layers (also implemented as convolutions)

This CNN has been designed as a slight simplification over the AlexNet architecture. It takes as input color images of 224 × 224 pixels (same as our samples). All layers are followed by ReLU activations. Some convolutions (S) are strided. A final softmax computes the class probabilities. The idea is that in deep CNN only the last layer is problem specific, while the previous layers recognize general characteristics of an image that can be exploited for different analysis.

3 Model

In terms of low level features for what was said previously in section 1.2 regarding the structure of the photos we will analyze two different pairs of low level features:

- **Color low level features**
- **Direction low level features**

Comparing the test and train results obtained from a simple neural network having as input the size of the features and at the output a number of output neurons equal to the classes (three) with the following parameters : *lr = 0.0001*, *steps = 3000*, *batch size = 50* the results obtained were:

Low level Features	Test	Train
Color Histogram	57.9%	57.3%
RGB co-occurence matrix	59.6%	52%
Edge direction histogram	71%	68%
Co-occurrence matrix	59%	56%

Looking at these results I combine the best low level features for each category. In particular, color histogram and edge dir histogram, the result obtained was 76% accuracy in train and 74% accuracy in test.

To increase the accuracy of the previously selected combination we can perform a normalization on the data, the normalization algorithms considered were three:

1. **L1** 45% accuracy on train , 46% accuracy on test. In this case i have worst performance than before.
2. **L2** 74.8% accuracy on train , 74% accuracy on test. In this case i decrease the overfitting but not the performances.
3. **Mean and std** 99% accuracy on train , 96% accuracy on test. In this case I have significantly increased the performances.

It seems evident that the best normalization choice is the average with the standard deviation.
As for the neural features extracted from ppp in just 1000 ephocs we obtained 100% on test and 99.3% on train in only 1100 iterations.
Using the validation data, several models were tested:

- **KSVM**: Using multiclass KSVM with one vs one methodology (since we have a small number of total classes) the following 2 models have been trained:
    1. **KSVM with low level features**: the main parameters used were the following: *Radial kernel function*, **kernel param = 0.01** and *1000 steps*
    2. **KSVM with neural features**: the main parameters used were the following: *Radial kernel function*, **kernel param = 0.03** and *1000 steps*
- **MLP without hidden layers** divided in :
    1. **MLP with low level features**: with *lr = 0.0001*, *steps = 1000*, *batch size = 50*
    2. **MLP with neural features**: with *lr = 0.0001*, *steps = 3000*, *batch size = 50*
- **MLP with an hidden layers (128)** divided in :
    1. **MLP with low level features**: with *lr = 0.0001*, *steps = 1000*, *batch size = 50*
    2. **MLP with neural features**: with *lr = 0.0001*, *steps = 3000*, *batch size = 50*

4 Conclusion

The result are the following:
Looking at the results, it can be seen that except for the SVM with low level features all the other

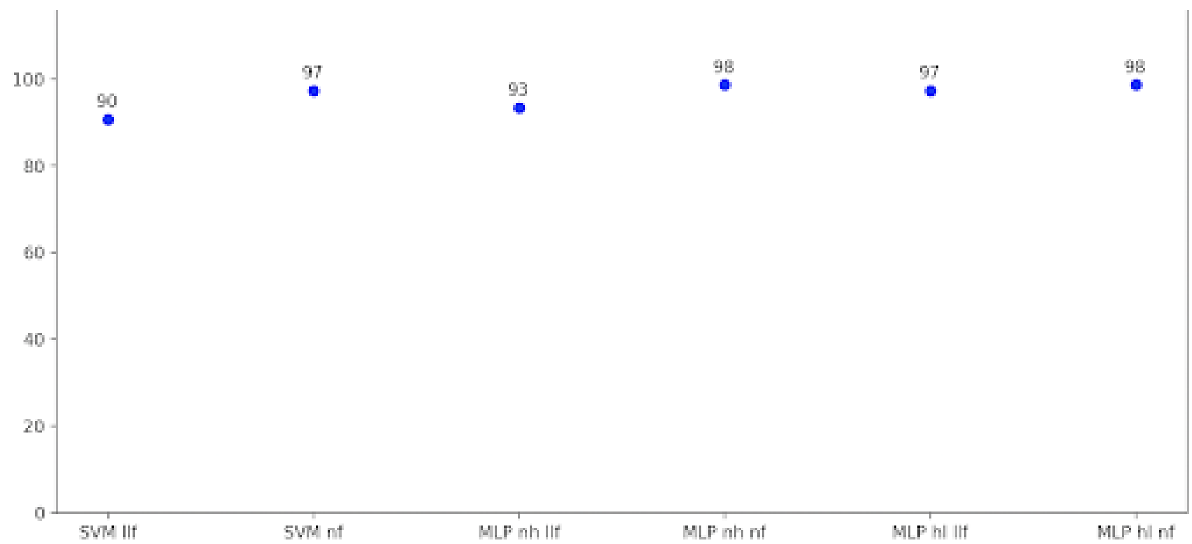

Figure 3: Scatter plot with accuracy of each model on validation data

models give fairly equivalent results, so the choice of the best model depends on various situations and

various factors, so I think making a compromise between robustness in the variation of samples and accuracy that the most suitable model for this type of classification is the neural network without hidden layer with the neural features.
However, looking at the confusion matrix in Figure 4 we see that the model makes some mistakes. in

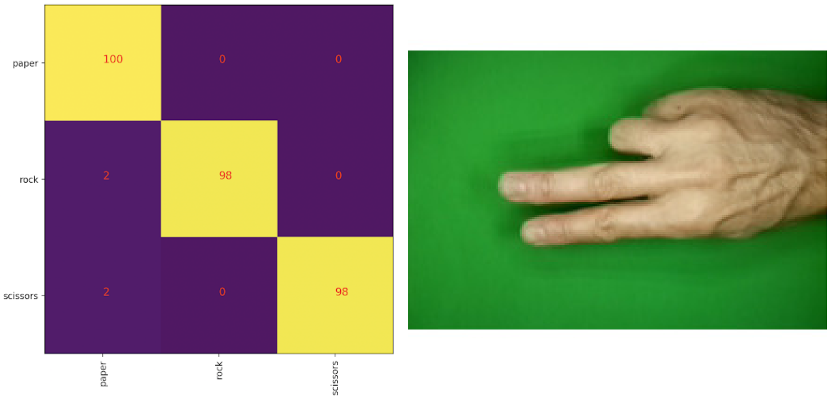

Figure 4: Confusion matrix MLP nhl with neural feature and Missclassified image

particular the rock and scissors class are confused in 2% of cases with the paper class and this, as seen in the next image, starts with a high probability. This image has been miss-classified with a probability of 81.9% probably because the shape of the hand forming the scissor is unusual and carried out with the last 2 fingers, which are always present in the shape of the paper, this can lead the classifier to make mistakes.

”*I affirm that this report is the result of my own work and that I did not share any part of it with anyone else except the teacher.*”