Scortex project:

Mohcine Heddi

1 Introduction:

In this project, we study a classification case.
The classification task concerns chars74K database. The first part of the report introduces the approach that we used to preprocess our database, and the second part concerns the classification algorithm that we used.
Kindly find enclosed to this report, the python scripts.

2 Data preprocessing:

For this application, we are trying to classify 62 different classes compose of letters and numbers: 0-9, a-z and A-Z.

2.1 Train and test set:

Given the small size of the database, we did not use a valdation set, db.pickle is a pickle file that contains a dictionary where you can find the split between the train and test dataset. 70% for the training set and 30% for the test set.

2.2 data preprocessing:

Given that the images have different sizes, the first preprocessing step was to resize all the images, we used a 28*28*3 size. Our classification task is not linked with the colors in the images, so we converted all the RGB images to gray images. Then we normalized them. Please find the python function in utils.py file.

2.3 Data augmentation:

As described in the next section, we generate features using a neural net. To perform this task, we have duplicated our training set by rotating the images, we used 2 differents angles: 90 and 180 as described in create_tfrecord.py file that creates the tfrecord.

3 Classification task:

The classification task is divided into 2 main steps: feature generation and classification.

3.1 Feature generation:

To generate features, we used a deep autoencoder with the following details for the encoder:
- input size: 784
- first hidden layer: 1000
- second hidden layer: 500
- third hidden layer: 250
- fourth hidden layer: 30

The decoder is the symmetric of the above layers.

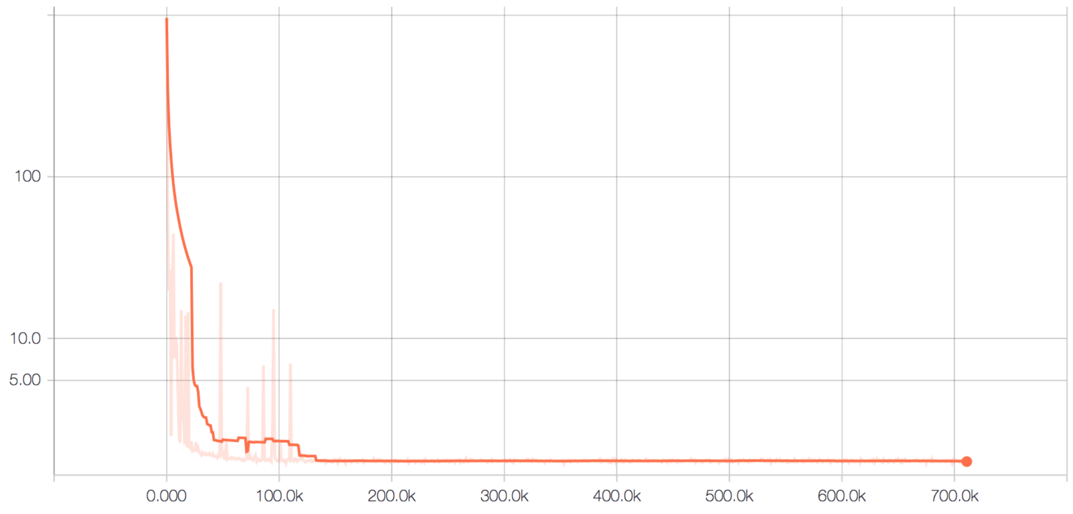

Figure 1: loss function of the autoencoder

3.2 Classification task:

We propose to train a SVM on the hidden codes of the training dataset. The training is taking too much time. Kindly find the code enclosed with this report.

4 Conclusion:

Given the short time that we have, the SVM did not finish its training. Once the training done, we can evaluate the classification task using the following metrics:
- accuracy

-top 1 and top 5 error rates
- ROC curve for multi-classes: we can manage that by comparing each class Vs all the others

Another approach would have been to train a CNN over the training dataset and perform the classification task with a softmax function at the end of the fully connected layer (62 neurones as we have 62 classes).