Kaggle Project Report

Kernel Methods for Machine Learning

Émile Enguehard, Alex Auvolat

1 Task presentation

The dataset is composed of a subset of the MNIST dataset, slightly altered: the training set is composed of 5000 labeled MNIST digits (28x28, ie 784 features) which have been mirrored/rotated. Here are a few digits from the dataset:

Figure 1. A two

Figure 2. A nine

Figure 3. A five

Figure 4. A four

As we can see, the digits are hard to recognize for a human. However the dataset has regularities that makes it possible to learn a good classifier that is adapted to this dataset.

The test set, which is composed of 10000 digits, must be classified as precisely as possible. The score used for the Kaggle ranking is the proportion of correctly classified digits.

2 Feature extraction

We first convolve the images with a series of Gabor filter at six different orientations. The equation of a real Gabor filter which we used is:

$$\begin{aligned} g(x,y;\lambda,\theta,\psi,\sigma,\gamma) &= e^{-\frac{x_\theta^2+\gamma^2 y_\theta^2}{2\sigma^2}}\cos\left(\frac{2\pi x_\theta}{\lambda}+\psi\right) \\ \begin{pmatrix} x_\theta \\ y_\theta \end{pmatrix} &= R_\theta \begin{pmatrix} x \\ y \end{pmatrix} \end{aligned}$$

The filters we used were of size 11×11. The parameters were defined as follows:

$$\begin{aligned} \lambda &= 5.942 \\ \theta &= \frac{2k\pi}{6}, k = 0...5 \\ \psi &= -0.556 \\ \sigma &= 2.239 \\ \gamma &= 1.386 \end{aligned}$$

We then calculate for each convolved image (each corresponding to a certain filter orientation) the mean absolute value over patches of 4×4. This gives a set of $\frac{28}{4} \times \frac{28}{4} \times 6 = 7 \times 7 \times 6 = 294$ features, instead of the 784 original pixels of the image.

We also do two kinds of normalization to help the classifier work: we normalize all the images independently before applying the Gabor filters, and we center each of the 294 features relatively to the whole dataset after doing the filtering and the pooling.

3 Classification

The classifier we use is a standard kernel C-SVM, with a RBF kernel. The RBF kernel is defined as follows:

$$K(x_i, x_j) \;=\; e^{-\gamma \|x_i - x_j\|_2^2}$$

The value of γ we used is $\gamma = 0.7357$.

A kernel C-SVM classifier is a function defined by:

$$\hat{f}(x) \;=\; ...$$

where $\hat{w}$ minimizes:

$$...$$

The parameter C that we used is $C = 263$.

We implemented the C-SVM by transforming this problem into a quadratic problem, which we solve using the `cvxopt` library for Numpy. We use a one vs. one voting strategy to handle the 10 classes of the dataset.

We also implemented a ν-SVM classifier to compare the results, but the C-SVM was found to perform better.

4 Hyperparameter search

The optimal parameters for our model were found using randomized parameter search with 5-fold cross validation. The parameters were first set by intuition to be bounded by some first intervals, that we refined manually when good parameters were discovered. The values for the parameters were chosen randomly in these intervals, uniformly on a linear or logarithmic scale depending on the parameters.

5 Our results

Table 1 shows our results with the best hyperparameter combination we found for the C-SVM and the ν-SVM. In all cases we use a Gaussian RBF kernel. We see that both models are very close, but the C-SVM was able to perform slightly better.

Model	**Cross-Valid. error rate**	**Public score**	**Private score**
Gabor filters + RBF ν-SVM	2.7%	0.9764	0.9720
Gabor filters + RBF C-SVM	**2.5%**	**0.9774**	**0.9734**

Table 1. Results for our method with two classifiers.