Machine Learning KIM.ML09 – Semester Project Suggestion "Handwritten digit recognition"

Overview. This is the archetypical supervised machine learning task: classifying handwritten digits. There is a handwritten digits benchmark dataset called the MNIST dataset (https://en.wikipedia.org/wiki/MNIST_database) which without doubt is the most widely used benchmark dataset in the history of machine learning, period. However, I would suggest that for this semester project you use an older, cleaner and much smaller dataset, the "Digits" dataset that I also used for demonstrations in Section 4 of the lecture notes – what you can learn by fighting with this dataset is not different from what you can learn when struggling with MNIST, but the turnover times per simulation / learning run are much faster with this smaller dataset, which will help you to explore more aspects and try out more tricks than you could with MNIST.

Data. The dataset that I suggest to use is the same as I use in the lecture notes – it has been first used in:

J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, *On combining classifiers*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(3):226– 239, 1998. (file 1950_KittlerDuin_pami_98_ccomb.pdf is attached)

This dataset contains 2000 grayscale images of handwritten digits, 200 from each class. The images are 15 × 16 sized, making for n = 240 dimensional image vectors. The data are in the attached text file `mfeat-pix.txt`, one vector per row, sorted such that the first 200 rows are "0" digit examples, the next 200 are "1" digit examples etc. The grayscale encoding in `mfeat-pix.txt` is done by integer steps from 0 (white) to 6 (black).

There is a standard split into a training and a testing set: the first 100 examples per class are to be used for training, the remaining ones for testing.

The attached Matlab file basicDemo.m reads in the data from `mfeat-pix.txt`, creates some plots (as in the lecture notes), and learns and tests a very simple (low-performing) linear classifier. Python users will easily translate the Matlab code.

Difficulty. Like all ML tasks, there is no upper limit for tricks and ingenuity that one may invest. However, classifying handwritten digits on the basis of a small and clean dataset can be called a relatively simple and straightforward problem.

Hints and comments.

- The best published results on this dataset achieve a test error rate that is slightly better than 2%. If you achieve something better than 5% you are doing ok., if you come close to 2.5% or even better you have achieved a seriously well-done job in a short time.
- This dataset is small (by deep learning and other standards), which means there is a big danger of overfitting and a necessity to do a careful regularization. Very good for experiencing the essential nasty core of statistical modeling!
- With purely linear methods you don't reach far in this task. But, nonlinear feature extraction followed by a linear regression can give good results. A wide playground for ingenious feature designs! Or, you can use nonlinear classifiers, for instance based on neural networks. Convolutional neural networks (CNNs) would be the typical choice in today's deep learning world. Because of the small size of the training dataset this would need careful regularization to prevent overfitting. The best reported results in the literature (slightly better than 2% test error rate) were achieved long time ago, before the rise of CNNs. Would be interesting to see whether with CNNs one can reach < 2% error rates with relative ease.