Introduction

The big data regime posits a basic assumption of data abundance. What should we do when we have a lot of training data, but also a lot of parameters to learn? We already know this hurts regression tasks. Overfitting is a major concern.

The project explores this high-dimensional setting for supervised digit recognition on MNIST data set, containing $28 \times 28$ images. We test the performance for different training sizes $m = 100, 250, ..., 10000$, and draw empirical conclusions for k-nearest neighbor (k-NN), multi-class logistic regression (MLR), SVM and (small, relatively deep) convolutional neural network (CNN).

Overall: CNN beats all

We test several representative algorithms, without regularization.

SVM: $\ell_2^2$ regularization is (still) very helpful

We test linear SVM with $\ell_2^2$ regularization. Smaller $C$ imposes heavier $\ell_2$ penalization. Oddly, SVM should be insensitive to overfitting...

PCA may not be a terrific idea

We test linear SVM with PCA preprocessing for $m = 750$. Had PCA worked, we should have seen a clear drop in error rate at some intermediate dimension.

CNN: (Un)predictable

We test several CNN architectures. We list below the structure of convolutional layers, where C1, C3, C5 refer to layers of original LeNet-5, and 20 refers to a layer with 20 maps (same for 40, 60, 120). In-between subsampling layers and the last fully connected layer are not shown. All layers have trainable weights.

We test their performances without regularization.

Note that even with $m = 10000$ or more, we are still in the regime # training data $\ll$ # parameters. Observations:

- Changing the training size may alter the relative performance.
- Reducing the number of trainable weights may help when there is less data. But not necessarily.
- Increasing depth may not help.