Convolutional Neural Networks for Classifying Handwritten Digits

Favyen Bastani
fbastani@perennate.com

Zhengli Wang
wzl@mit.edu

December 5, 2014

1 Introduction

Image recognition algorithms have become pervasive: on Google Maps Street View, image recognition protects privacy by automatically blurring faces and license plate numbers [1]; robots use object recognition to carry out various tasks; meanwhile, individuals routinely employ optical character recognition and facial recognition. Over the last decade, image recognition approaches have changed dramatically. Initially, feature extraction techniques such as HOG and FAST were typically combined with support vector machine (SVM) classifiers, with research often focusing on selection of SVM kernels and defining features for specific applications. Recently, though, neural networks have received renewed attention and achieved high classification performance.

Cells in the mammalian visual cortex are able to identify local patterns in the visual field [2]. Convolutional neural networks (CNNs) aim to do the same through *convolutional and pooling layers*. While traditional neural networks are composed of densely connected layers, where each neuron in the previous layer connects to each neuron in the next layer, convolutional layers restrict filters to be applied on local sub-regions of an input image. By training these layers through back-propogation, these filters eventually detect specific features in the image that help with classification.

In this project, we implement a fast CPU-based CNN in Python using the `numpy` scientific computing library, and apply it to the MNIST handwritten digits database [3]. We evaluate the performance of various neural network parameters (including layer configuration, output layer activation function, and training algorithm) in terms of classification accuracy and speed, and compare CNN to feature extraction approaches with SVM. We find that ...

In Sections 2 and 3, we detail convolution and pooling functions, the convolutional neural network structure and training algorithm, and our implementation. In Section 4, we experiment with various neural network parameters, and in Section 5, we compare with SVM-based algorithms.

2 Convolution and Pooling

3 Convolutional neural networks

4 CNN evaluation

5 Comparison with SVM-based recognition

6 Conclusion