Tooling around with MNIST: what's the best model architecture?

Alec J. Hoyland
Center for Systems Neuroscience
Boston University
Boston, MA 02215
ahoyland@bu.edu

Abstract

This is the abstract.

1 Introduction

Optical character recognition (OCR) is the ability to take as input an image including handwritten or printed text, and output the sequence of characters contained within the image. OCR engines have many applications, including data entry (e.g. for checks, passports, invoices), book-scanning (e.g. for Project Gutenberg), pen computing, and in assistive devices (e.g. for blind and visually-impaired individuals). In an OCR engine, text is read from a scanned document or image, and translate the images into a form manipulable by the computer, such as plain text, enabling a multitude of other analyses and operations. As the saying goes, "You can't grep dead trees!"

A successful model is one which accurately predicts the correct label for the image (i.e. recognizes the digit), and does so quickly. A model that fails to accurately label is not useful, as is one that labels accurately, but too slowly to be useful in real-world applications.

We describe two measures of efficiency, training time and testing time. The former is the total runtime of the training phase, and the latter is the total runtime of the testing phase. Since the model need only be trained once (or perhaps fine-tuned later), minimizing runtime on the testing set is more important, since in a real-world scenario, the "testing" set will be monumentally larger, consisting of any documents or images the model is used to annotate.

Intuition indicates that simpler models, which involve less computationally-involved operations, will perform better on measures of efficiency, though it is possible that the decrease in accuracy would be unacceptable.

In this paper, we will explore a convolutional neural network and a feedforward neural network, with various training schedules, and benchmark accuracy and runtime speed.

2 Related Works

The MNIST dataset [1, 2] has been used as a standard machine learning benchmark for more than two decades. It is comprised of 60,000 28x28 grayscale images of handwritten digits 0-9. The handwriting samples themselves were written by American high school students, and American Census Bureau employees. While concerns about the construct validity of the dataset have been raised in recent years [3] the dataset and its variants remains a staple of machine learning benchmarking, and serves well for evaluating the quality of various model architectures.

Preprint. Work in progress.

The MNIST dataset is suited to a classification task, where the image is read as input, and the output is the numerical digit label (i.e. 0-9). Many different model architectures have been implemented to achieve highest accuracy on the classification task.

Many different model architectures have been tested against the MNIST dataset. Linear classifiers comprise some of the simplest of models, and traditionally fare poorly on digit identification [1]. Some of the first attempts based on multilayer neural networks with backpropagation [1, 4], were performed by the original curators of the MNIST dataset. Many variations have been tested, including limited receptive field models [5], and convolutional neural networks [6]. The latter exploits elastic training image deformations, which achieved state-of-the-art accuracy. Image deformation has also been explored in hidden Markov models [7]. Since then, methods have improved in GPU-based computations in deep, big simple neural networks [8, 9]. Support vector machines [10] have also been used for image-recognition tasks.

3 Methods

In this paper, eight models are implemented and compared to each other. The goal is to determine what sort of basic model architecture performs the best, in efficiency of training/evaluating, as well as in accuracy of the resultant classification. We consider a convolutional neural network and a fully-connected feedforward neural network, with modifications to the loss function and training schedule.

3.1 Convolutional neural network

The convolutional neural network (CNN) consists of three convolutional layers, a fully-connected layer, and a softargmax output.

The first convolutional layer operates on a 28x28 image, with a 3x3 window, padding of (1, 1), and a ReLU activation. This leads to 16 units in the second convolution layer, which operate with a 3x3 window and padding of (1, 1) on 14x14 images. This convolutional layer passes through ReLU activation to a third convolution layer, which acts on 7x7 images, with the same padding and ReLU activation as before.

Each convolutional layer is separated by a 2x2 maximum pooling layer.

Finally, the 3-dimensional tensor is flattened to 2 dimensions, and passed to a 288-unit fully-connected layer with linear activation, and 10 outputs (corresponding to the digits). The softargmax function is used to squash the outputs to normalized probabilities.

Loss is computed by the cross-entropy function, and the Adam optimizer with $\beta = 0.001$ is used for training [11].

3.2 Fully-connected neural network

A multi-layer neural network was also tested. This network consists of 2 fully-connected, feedforward layers each with 32 hidden units. ReLU activation is used between the hidden layers, and the softargmax function is used to squash outputs.

Cross-entropy loss is used, and Adam is the optimizer algorithm.

3.3 Modifications to existing models

Several alterations were made to the base models to attempt to improve accuracy and efficiency.

First, the input data were fuzzed with Gaussian noise ($\xi = 0.1$) to try to make the model more robust. Second, an modified training schedule was implemented, with early-stopping at 99.9% accuracy to prevent overfitting, and an adaptive learning rate.

In the adaptive learning rate schedule, if no improvement was seen in 5 epochs, the learning rate was decreased by 90% to a minimum of 10^{-6}. If no improvement was seen after 10 epochs, training stops early, as the model is assumed to have converged to a local minimum.

These modifications result in a total of eight models evaluated:

Table 1: Results of simulations

Model Name	Training Time (s)	Testing Time (s)	Testing Accuracy
CONV	320	3.72	0.985
CONVFUZZ	314	3.42	0.984
CONVADPT	336	3.78	0.987
CONVFULL	376	3.38	0.986
DENSE	67.9	0.053	0.906
DENSEFUZZ	144	0.055	0.901
DENSEADPT	9.51	0.058	0.116
DENSEFULL	0.114	0.103	0.117

- CONV, the base convolutional neural network
- CONVFUZZ, CNN with input fuzzing
- CONVADPT, CNN with adaptive learning rate
- CONVFULL, CNN with both input fuzzing and adaptive learning rate
- DENSE, the base fully-connected feedforward neural network
- DENSEFUZZ, feedforward network with input fuzzing
- DENSEADPT, feedforward network with adaptive learning rate
- DENSEFULL, feedforward network with both input fuzzing and adaptive learning rate

3.4 Evaluating quality of models

Models were evaluated for accuracy and efficiency.

Accuracy is defined as the number of correct labels divided by the size of the test set. Efficiency was measured by training time, and testing time. Training time is defined as the amount of time for which the model is trained. Testing time is the amount of time it takes for the model to label the testing set.

Unless training time is prohibitively long, testing time is more important. This is because possessing a model that can identify characters and glyphs quickly, is important for real-world applications of optical character recognition. A model does not need only to be accurate, it should also be fast.

4 Results

Models were run on a 4-core cluster computer equipped with an NVIDIA Tesla K40m GPU. Results are tabulated below:

5 Discussion

6 Conclusion

References

[1] L. Bottou, C. Cortes, J.S. Denker, H. Drucker, I. Guyon, L.D. Jackel, Y. LeCun, U.A. Muller, E. Sackinger, P. Simard, and V. Vapnik. Comparison of classifier methods: A case study in handwritten digit recognition. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 3 - Conference C: Signal Processing (Cat. No.94CH3440-5)*, volume 2, pages 77–82 vol.2.

[2] Handwritten digit database.

[3] Chhavi Yadav and Léon Bottou. Cold Case: The Lost MNIST Digits.

[4] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. 86(11):2278–2324.

[5] Ernst Kussul and Tatiana Baidyk. Improved method of handwritten digit recognition tested on MNIST database. 22(12):971–981.

[6] Patrice Y. Simard, Dave Steinkraus, and John Platt. Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis.

[7] Daniel Keysers, Thomas Deselaers, Christian Gollan, and Hermann Ney. Deformation Models for Image Recognition. 29(8):1422–1435.

[8] Dan Claudiu Cirean, Ueli Meier, Luca Maria Gambardella, and Jürgen Schmidhuber. Deep, Big, Simple Neural Nets for Handwritten Digit Recognition. 22(12):3207–3220.

[9] Dan Cirean, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3642–3649.

[10] Dennis Decoste and Bernhard Schölkopf. Training Invariant Support Vector Machines. 46(1):161–190.

[11] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization.