MNIST Image Classification with Convolutional Neural Network

Jason Rich
Old Dominion University
Computer Science
Norfolk, Virginia 23504
jrich069@odu.edu

***Abstract*—In this article, I will demonstrate the use of a Convolutional Neural Networks (CNN) as a technique for image classification. The dataset used for this study is The MNIST database of handwritten digits, which contains a training set of 60,000 examples, and a test set of 10,000 example. The dataset is a subset of the larger set available from National Institute of Standards and Technology [1]. The goal of this paper is to show that analyzing the MNIST data, using Anaconda's python 3.5 distribution, and Google's TensorFlow package for python3, on a standard laptop is not only possible, but also efficient, accurate, and certainly affordable. Moreover, I will show that CNN will converage in as little as 2000 steps, and that as the steps increase, the error rate draws closer and closer to zero, as the accuracy of the model grows closer and closer to 100%.**

I. Introduction

Historically, to preform image processing, whether high quality, digital examples, or hand written notes, presented using a standard office scanner, the machine learning practioner would have to extract language dependent features like curvature of different letters, spacing, black and white letter, etc., only to use a classifier such as Support Vector Machine (SVM) to distingish between writers [2]. With the publication of (LeCun et al. 1998), the analysis of handwritten, variable, 2D shapes with Convolutional Neural Network was shown to outpreform all other techniques [1].

I will show that given the advance in Application Program Interface frameworks, such as TensorFlow [3], Keras [4], H2O [5] as-well-as others, have provided not only machine learning researchers and practioners the ability and tools to quickly and efficiently analyze larges amounts of data, with what are traditionally thought of as mathematically complex, but also overly expensive, both runtime and monetarily.

The key observation in this study was, given a well studied dataset, and an evolving deep learning algorithm, the ability of personal hardware, in my case my 2011 Mac Book Pro, with 16GB of RAM, a 1TB hardware, and an i5 Intell processor, to reproduce results originally calculated on academic or remote research servers. This says a lot about the hardware, but more so about the work, research, and improvements that have rollup into the current versions of modern day deep learning algorithms.

Hopefully, by the conclusion of this paper, I will have shown, that we have come a long way the field of deep learning. However, I also hope to show thar we have much more work remaining, and efforts in the fields of quantum machine learning, quantum deep learning, and continued improvment in high performance computing, are quintessential to furhter the advancements, demonstrated within this paper.

II. Related Work

A. Foundational Work

LeCun et al. (1998) laid the foundation ground work for all current convolutional neural network architecture and image processing, building on the concepts of Gradient-Based Learning. The work of LeCun et al. (1998), and others, set the tone for work that is happening today. Without the work of people like LeCun, Hinton, and Ng, we may not have the bleeding edge algorithms or the tools to analyze the data we can today.

B. Gradient-Based Learning

The general problem of minimizing a function with respect to a set of parameter is at the root of many issues in computer science. Gradient-Based Learning draws on the fact that it is generally much easier to minimize a reasonably smooth, continuous fucntion than a discrete (combinatorial) function. This is measured by the gradient of the loss function with repect to the parameters. Efficient learning algorithms can be devised when the gradient vector can be computed analytically (as opposed to numerically through perturbation). Furthermore, LeCun et al. (1998) notes; …the basis of numerous gradient-based learning algorithms with continuous-valued parameter. In the procedure described continuous-values parameters W is a real-valued vector, with respect to which $E(W)$ is continuous, as well as differentiable almost everywhere. [T]he simplest minimization procedure in such a setting is ther gradient descent algorithm where W is iteratively adjusted as follows:

$$W_k = W_{k-1} - \epsilon \frac{\partial \mathbf{E}(W)}{\partial W}$$

In the simplest case, ϵ is a scalor constant [1]. Moreover, LeCun et al. (1998) note: A poplar minimization procedure is the stochastic gradient algorithm, also call the the on-line update. It consists in updating the parameter vector using a noisy, or approximated, version of the average gradient. In the most common instance of it, W is updated on the basis of a single sample:

$$W_k = W_{k-1} - \epsilon \frac{\partial \mathbf{E}^{p_k}(W)}{\partial W}$$

With this procedure the parameter vector fluctutates around an average trajectory, but usually converages considerably faster than a regular gradient descent and second order methods on large training set with redundant sample...[1]. For more information on stochastic gradient descent models see Bottou (2010) and Sutskever et al. (2013).

C. Image Processing

However, with the advent of more sophisticated digital carmers, with great pixel quality, and pixels pre-inch, images become larger and larger. The traditional methods of image classification, using a fully-connected network, with hundreds of hidden units in the first layer [1], [7], [8], creates thousands of weights. Furthermore, using a fully-connected network negates the fact that neightboring pixels are more coorelated that non-neighboring pixels [7].

The primary advantage of using a convolutional neural network is the convolution itself. Convolutional neural networks are specifically designed for processing data that has a know grid-like topology [8]. Image data, as noted in Goodfellow, Bengio, and Courville (2016), should be thought of as a 2-D grid of pixels. I will provide a brief summary of convolution in section III, as well as the key differences in machine learning and deep learning.

III. Convultional Neural Net

A. Convolution:

$$S(t) = \int x(a)w(t-a)da$$

, annotated another way:

$$S(t) = (x * w)(t)$$

B. Deep Learning:

IV. Experiment

A. Dataset

The dataset used for the study in the MNIST [3], extracted using TensorFlow. The dataset used for this study, is a subset of a much larger dataset, orignally made available by NIST [1]. It consist of 60,000 images for training the models, and 10,000 images for testing the models.

The images in the dataset were pre-processed and stored as a greyscale, centered $28x28$ fixed-size image. The pre-processing performed on the images, greatly improves the algorithms ability to process the data, thus assisting in minimizing the error rate.

Other than the image files, the dataset also includes the label for classifying the images. The values of the labels are on a range from 0 to 9. The image training dataset is approximately 0.099 gigabytes and the image testing dataset is considerably smaller.

The dataset was pulled locally using the`tensorflow.examples.tutorials.mnist`module, and calling`input_data`funciotn with one hot encoding.

I will fully explain the code in the next subsection.

B. Code

C. Results

V. Conclusion and Future Work

The conclusion goes here.

Acknowledgment

The authors would like to thank...

REFERENCES

Bottou, L. 2010. “Large-Scale Machine Learning with Stochastic Gradient Descent.” In *Proceedings of 19th International Conference Computer Statistics*, 177–86. Princeton, NJ: Springer.

Goodfellow, I, Y Bengio, and A Courville. 2016. *Deep Learning*. 1st ed. Cambridge, MA: MIT Press. http://www.deeplearningbook.org.

LeCun, Y, L Bottou, Y Bengio, and P Haffner. 1998. “Gradient-Based Learning Applied to Document Recognition.” In *Proceedings of the IEEE*, 86:2278–2324. 11. http://yann.lecun.com/exdb/publis/pdf/cox-98.pdf.

Sutskever, I, J Martens, G Dahl, and G Hinton. 2013. “On the Importance of Initialization and Momentum in Deep Learning.” In *Proceedings of the 30th International Conference on Machine Learning*, 1139–47. ICML.