OPTICAL RECOGNITION OBJECTIVES

EXECUTIVE SUMMARY

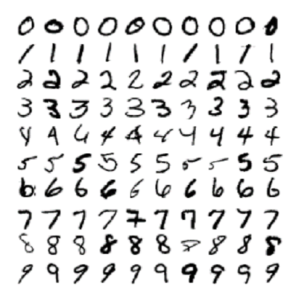
In financial services, one of the most daunting challenges facing the industry is embracing the seismic shift towards the analytical revolution. This has the potential for enormous impacts to the everyday jobs of people spanning various lines of business from the retail banking, mergers & acquisitions as well as sell-side investment banking. An example of such a technology is optical character recognition, which is a foundational aspect in the domain of computer vision.

Optical character recognition is a data analytics technology that is used to convert digital representations of characters and symbols, like those that would appear in a scanned document, to its native digital representation in the computer. Once the data is in the native representation of the computer, further automation and analysis can be conducted seamlessly across the organization.

For example, this technology could be used to automate the scanning and process of a deposit check for a bank teller or to scan and classify thousands of documents made available via a data room during a merger or acquisition. Having the ability to automatically digest vast amounts of human-produced content and transform it for digital and analytical processing would give any financial organization a substantial competitive edge in practically every area of its daily business and operational process.

RESEARCH DESIGN

For this problem we used a well-known, universally recognized and accepted database of hand-written digits composed of a series of 70,000, 28x28 pixelized, images that also contain an associated label that specifies the images true digit value. This database has been used in countless research studies, machine learning competitions (such as [Kaggle](https://www.kaggle.com)), and academic classrooms throughout the world.



Our classification system uses the first sixty-thousand images in the database as a training dataset and the last ten-thousand as our test and validation dataset. We will take the same sixty-thousand images and classify them with different approaches.

The first approach we will take is to establish our baseline results by using a multilayer perceptron model with simplistic tuning parameters on the trading set. The results from our initial run of the model will serve as a baseline for the training performance/cost, training accuracy, and testing accuracy. The baseline version and the research/test versions of the model will all use the same weights and biases to maintain consistency throughout the testing so that we can lock in on the parameter tuning.

Once the benchmark case is solved for regarding computational cost vs. modeling accuracy and training-time, we will look to improve on my tuning the key parameters of the model. The learning rate, which controls how quickly the model converges on the local minima, the number of training epochs (iterations), and the batch size which determines how many samples will get propagated through the network.

TECHNICAL OVERVIEW

The model construction and benchmarking methods were conducted purely in the cloud, leveraging a pre-canned environment from a world-class provider of machine learning solutions, Google, called Colabratory. This cloud environment enables us to not only research more efficiently, while also allowing maximum reproducibility by taking out the variability of an individual desktops hardware configuration for the benchmark results. Additionally, we can push our research globally and allow any astute reader to reproduce our results for themselves in a sandboxed environment.

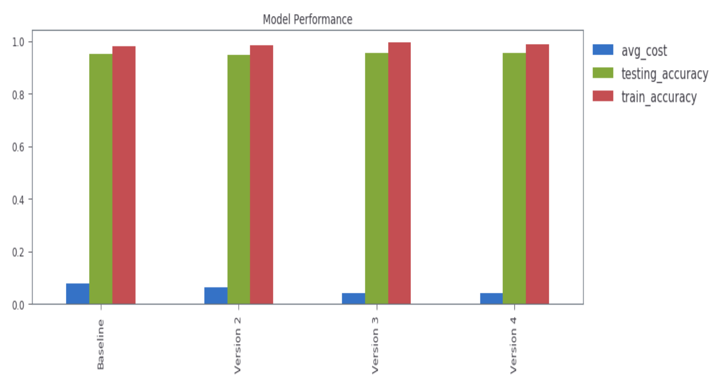
Additionally, for this research we also leveraged an industry-leading machine learning framework, TensorFlow, which Google also produces. TensorFlow gives us access to the same underlying technology that powers several of the most advanced analytical systems in production today.

Using this framework, we set up an artificial neural network algorithm for training our classification system. For this problem, we chose a Multilayer Perceptron (MLP), an algorithm that is trained on the MNIST dataset with various parameters so that we can research the optimal training configuration, in which we reach maximum accuracy with minimal time spent on the algorithms I/O and computational heavy training phase.

At the heart of this study is the Multilayer above Perception algorithm, and three tuning parameters: learning rate, training epochs, and batch size. We will execute the "baseline" algorithm with the minimal specifications to get the algorithm to perform substantially better than the flip of a coin, which would be our worst case regarding performance, as we would have simply wasted resources for absolutely no benefit. From there, we will tune the parameters based on a training time cut-off and attempt to find the equilibrium are of maximum performance in minimal time.

CONCLUSION

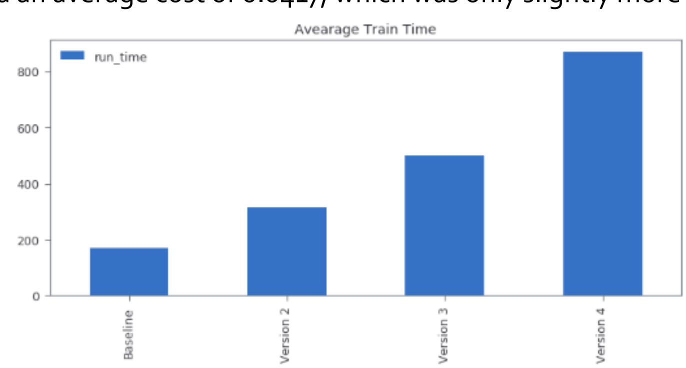
After completing our trials and research, we should first define how we measured each characteristic of the model. For accuracy, we used the TensorFlow 'accuracy' metric, which is the frequency with which the model's prediction matches the test label.



The baseline model has a testing accuracy score of .9491, compared to the most robust model, version 4, which has a testing accuracy of .9536, which are both outstanding scores.

We should note that the time sacrifice to get such a small increase in testing accuracy, the model training time went up by a factor of 2x, which is not outlandish. However, it is not ideal when building for large-scale systems.

It turns out there is a nice equilibrium area with the various combinations of parameters used to build this model. In the third version of the model, we used a learning rate of 0.015, with 500 training epochs and the same batch training size. This version yielded over 95% accuracy, .9544 to be exact, and an average cost of 0.0417, which was only slightly more than the base case version.



Using the methodology of parameter turning we have chosen for this research project, it is evident that the model accuracy scales linearly in training time for only a fractional increase in testing accuracy. Overall, I would recommend that if we are trying to solve for minimal training cost, the baseline version of this model is more than adequate regarding accuracy, especially relative to training time and resource intensity.

This research is by no means exhaustive or complete. We have merely scratched the surface on what is possible here with TensorFlow and some training data.

For additional information on how this research is conducted, please visit the Colabratory Notebook.