Capstone Proposal

Joel S

Machine Learning Engineer Nanodegree December 29th, 2016

Handwritten Digit Recognizer using PCA and KNN

Handwriting Recognition

Handwriting recognition is the ability of a computer to receive and interpret intelligible handwritten input from sources such as paper documents, photographs, touch-screens and other devices. The image of the written text may be sensed "off line" from a piece of paper by optical scanning (optical character recognition) or intelligent word recognition. Alternatively, the movements of the pen tip may be sensed "on line", for example by a pen-based computer screen surface, a generally easier task as there are more clues available [1].

Handwriting Recognition plays an important role in storage and retrieval of typed and handwritten information. A wealth of historical papers exists in a physical format. This includes genealogical information, old family records, written manuscripts, personal diaries and many other pieces of a shared past. Consistent review of this information damages the original paper and can lead to physical data corruption. Handwriting recognition allows people to translate this information into easily readable electronic formats [2].

Since 2009, the recurrent neural networks and deep feedforward neural networks developed in the research group of Jürgen Schmidhuber at the Swiss AI Lab IDSIA have won several international handwriting competitions. In particular, the bi-directional and multi-dimensional Long short-term memory (LSTM) of Alex Graves et al. won three competitions in connected handwriting recognition at the 2009 International Conference on Document Analysis and Recognition (ICDAR), without any prior knowledge about the three different languages (French, Arabic, Persian) to be learned. Recent GPU-based deep learning methods for feedforward networks by Dan Ciresan and colleagues at IDSIA won the ICDAR 2011 offline Chinese handwriting recognition contest; their neural networks also were the first artificial pattern recognizers to achieve human-competitive performance on the famous MNIST handwritten digits problem of Yann LeCun and colleagues at NYU [1].

Most modern tablet pc and hybrid laptops includes an active stylus and display which allows the user to write and draw on the screen. It would be very useful for the users if they can recognize, derive meaning and store the handwritten text into digital text which can be used within computer and text-processing applications. This motivated me to take the first step of implementing a Handwritten Digit Recognizer.

Problem Statement

The goal is to predict the digit handwritten in an image. The dataset contains images of handwritten digits and their respective numbers which can be used to train and test the model. This is supervised learning problem – more specifically a classification problem. The model should be able to classify which number (0 - 9) is handwritten in the image. The model can be scored for its ability to predict the number correctly over large different test data and real data.

Datasets and Inputs

The MNIST database [3] of handwritten digits has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image.

The original black and white (bi-level) images from NIST were size normalized to fit in a 20x20 pixel box while preserving their aspect ratio. The resulting images contain grey levels as a result of the anti-aliasing technique used by the normalization algorithm. the images were centered in a 28x28 image by computing the center of mass of the pixels, and translating the image so as to position this point at the center of the 28x28 field.

One-time conversion of the MNIST dataset is required to convert the image data into text data of pixel-values and combine them with their respective label and finally store them in CSV file format so that it can be used easily in the code. There will be two CSV files train.csv and test.csv after the conversion which will be provided by me during the submission. The training data set (train.csv) and testing data set (test.csv), has 785 columns. The first column, contains the label which is the digit that was drawn by the user. The rest of the columns (28x28 pixels = 784) contain the pixel-values of the associated image.

Thus we can use the train.csv dataset to train the model with pixel-values of the handwritten image as features and its respective label as target. We can also test the model by evaluating it against the test.csv dataset.

Solution Statement

Given that the problem is a supervised learning problem and more specifically a classification problem, there are lot of algorithms available for training a classifier to learn from the data. The algorithms chosen for this projects are Principle Component Analysis(PCA) and k-Nearest Neighbor(k-NN).

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables i.e. it will translate the pixel values into new features that are less correlated with each other, called principal components [4]. The principal components form the new feature space for which a classifier will be trained and tested.

k-Nearest Neighbors algorithm (or k-NN for short) is a non-parametric method used for classification and regression. In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors [5].

k-NN classifier is trained over the principle components of each handwritten digit image as features and its corresponding label as target. When predicting, k-NN classifier chooses k nearest neighbors based on the principle components of the test handwritten digit image and chooses the most common number assigned to the k nearest neighbors as the final predicted number.

Benchmark Model

The benchmark model is chosen from one of the kernels of the Kaggle Digit Recognizer competition [6]. The benchmark model solves the same handwritten digit recognition problem mentioned in this project by using a Random Forest classifier algorithm. It uses the same dataset used in this project. Thus making it a perfect benchmark model for this project. The result of the benchmark model is the accuracy score of the model. The same result can be measured for our model by calculating the accuracy score.

Evaluation Metrics

Accuracy is measured as ratio between the sum of true positives and true negatives and dataset size.

$$accuracy = \frac{True\ positive\ +\ True\ Negative}{dataset\ size}$$

True Positive and True Negative here are the same because both are the sum of all truly classified handwritten digit image samples. In general, its the measure of correctness of the model. It's just the ratio between correctly classified samples to the overall samples. It ranges from 0 – 1, where 0 means poor performance and 1 means good performance.

Project Design

First, the dataset need to be processed and converted once to CSV files for easy implementation as discussed in the Dataset and Inputs section. The processed data is then loaded into python using panda library. The loaded data will be first explored and visualized using numpy and matplotlib library to understand the nature and distribution of the data. Exploring the data will help us in deciding how to approach and whether any preprocessing of the data is needed. Then preprocessing of the data is done if necessary. Once the data is ready, we implement the Principle Component Analysis algorithm to convert the pixel-values to principle components i.e. new feature space as discussed in the solution statement. Now, a k-NN classifier will trained on the principle components of the training dataset (train.csv) and evaluated using accuracy score against the principle components of the testing dataset (test.csv). Now we can fine tune the parameters of PCA and k-NN using the Grid Search algorithm to find the highest accuracy score possible. Then the results can be analyzed and compared with respect to the benchmark model to know the overall performance of the model.

References

1. https://en.wikipedia.org/wiki/Handwriting_recognition
2. http://www.ehow.com/list_7353703_benefits-advantages-handwriting-recognition.html
3. http://yann.lecun.com/exdb/mnist/
4. https://en.wikipedia.org/wiki/Principal_component_analysis
5. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
6. https://www.kaggle.com/romanroibu/digit-recognizer/random-forest-digit-classifier