Machine Learning Engineer Nanodegree

Capstone Proposal

Edward Storey
September 2nd 2019

Proposal

The following sections will outline my proposal for my Machine Learning Capstone Project.

Domain Background

The project that I am proposing is to investigate improved machien learning methods for recognising digits in real world google map photos of address numbers. There are many previous examples of digit detection, most notable in handwriting detection [1][2][3]. This particular project concentrates on the case of address numbers of real world buildings. This dataset The Street View House Numbers (SVHN) Dataset [4] takes images of building numbers from around the world and separates them into both separate digit images and the full multi-digit number images.

The recognition of real world address numbers is particularly useful in the application of Google Maps and other map software. As is the case with other machine learning applications there is simply too much data available, millions of images of door numbers are shown on google maps. With inaccurate street lengths and no easy way of identifying individual buildings and their addresses reading real life address numbers from the map images is a good solution to the problem. Personally I have on several occasions found it difficult to use Google Maps to accurately locate an address I am looking for. I am interested in solving this problem as I believe it affects me and many other people that use services like these on a regular basis.

Problem Statement

Number image recognition is already a well researched area of machine learning, with digit recognition in “real life images” a sub set of this research. Commonly real life recognition will be in the form of handwriting recognition, in this case I will be researching the less researched form of real life images from the open world. I will be taking a dataset of 33402 .png images taken from real world address on google images and contains 73257 separate digits.

The main hurdles in this problem come from the quality and variety of the images themselves. Not only do the images vary greatly in size but some images are clearer than others. Certain images are taken from different angles and distances making giving a high variation in the clarity of the address numbers. Another hurdle for the project to overcome is that the accuracy of the algorithm has to be at or almost at 100% whilst not succumbing to over fitting. If even one digit in an image is classified incorrectly the address is essentially useless. These issues occur in every image in the data set and designing an algorithm to overcome them will be challenging, as the digit recognition must be near 100% but simultaneously not succumb to over fitting.

Dataset and Inputs

The dataset that I am using for this project is The Street View House Numbers (SVHN) Dataset and is a Standford University open source dataset [4]. The dataset takes its images from house numbers from Google Street View images sourced from around the world. It consists of two multidimensional arrays of data, one of the full images and one of the cropped images. The array containing the full images also contains data on the positions within the image of each digit shown and what that digit it. The array containing cropped images separates out every digit in each image and normalises them into 32x32 squares. Both sets of formatted data are further split into three parts: train, test and extra. The dataset are currently in MATLAB file formats specifically .m and .mat. Part of this project will be to convert them to Python friendly formats.

This data set differentiates itself from other digit recognition data by containing solely images of real-world places. Previously the majority of work has been done on computer generated images or hand written digits. This data set aims to open up the possibility of even more unclear digits to be successfully. This application would be most useful for digital maps such as Google Maps and GPS navigation systems, but also has potential for real-time use through a camera whilst travelling.

This project aims to test new machine learning approaches to recognising the digits shown in this data set that have not been explored in previous studies. The dataset was created for use in the study “Reading Digits in Natural Images with Unsupervised Feature Learning” [5] and applies various machine learning techniques to recognise individual digits by manually extracting them from the original images. The initial dataset that contains whole images is only used as a reference in this study and the algorithm cannot recognise multi digit numbers. A second study has been applied to this dataset, “Multi-digit Number Recognition from Street View”, which uses deep neural networks to recognise whole numbers [6]. The authors of the second study found that their algorithm was more accurate and could recognise whole numbers rather than single digits. However, they found that the deeper their neural network was the more accurate the algorithm became, but the longer the training time became. Since close to 100% accuracy is needed for this application accuracy is the most important aspect, however to make this usable in the practical world it needs to be as efficient and fast as possible as well and neither approach fully achieves this aim.

Solution Statement

To solve this problem I have taken inspiration from how I believe a human would read multi-digit number. I theorise that when a person recognises a number they first recognise each of the individual digits and then compile them into the full number. For example the number 153 would be first seen as 1, 5 and 3. Once the viewer had recognised each digit individually they would then compile the number and recognise it as one hundred and fifty three. I aim to emulate this approach to full number recognition with my algorithm by having 2 separate learning algorithms recognising different aspects of the full number. The full algorithm will have a first stage machine learning model that will learn to recognise single digits based on cropped single digit layer, similarly to the Stanford paper [5]. I then plan to apply this trained algorithm to the full images by scanning cropped portions of each image and recording the likeliness of a certain digit existing in that portion of the image. An array holding the likeliness of a digit occurring at each point will then be the input to a second learning model which will learn to recognise multi digit numbers. The second algorithm will aim to emulate the deep neural networks approach [6].

Benchmark Model

There has been much prior research done on image recognition in the past and within that there are many papers studying digit recognition. Within this Machine Learning Engineer Nanodegree Program the subject of handwriting recognition is explored in the module Intro to TensorFlow. Then more broadly image recognition was explored in the assessment Dog Breed Classifier. Both of these course modules will be used to inspire and help develop this Capstone Project proposal. There is also many research papers to draw inspiration from with the subject of digit recognition having been research in many studies across the last few decades [1] [2] [3].

Mostly I will be using the two studies mentioned previously as my benchmark models for this [5] [6]. I will be using the idea of single digit recognition from the Stanford paper on real world digit recognition [5]. In this paper the researchers found that a k-means algoirthm gave the best results for single digit real world recognition, but also found promising results by using neural networks. I will test both approaches and use my own findings to develop the first model in my algorithm. For the second model, designed to recognise multi-digit numbers through the use a Deep Neural Network (DNN) as seen in [6]. In this paper the researchers used a DNN of multiple layer with hidden layers of 3,071 units each. I will attempt to create a smaller DNN with less units per layer as to increase the efficiency of the model. I believe this will be possible as my DNN will only have to compute the frequency of digits occurring and not the recognise the shapes of the digits themselves.

Evaluation Metrics

Due to the nature of the dataset calculating accuracy is a relatively simple task. Since the dataset contains the value of each individual data and the file name of the .png file it is associated with it all that is required is to is record how many full numbers have been recognised correctly and take a mean average where success is marked as a 1 and a fail is marked a 0. This techniques will work for the initial single digit model as well.

Both benchmark studies emphasise the need for close to 100% accuracy to rival human tests. The Stanford paper [5] estimates human performance as having a score of 98% accuracy on this dataset, I will use this as a benchmark for this project. 90.6% accuracy was achieved with the k-means model [5]. When DNNs were applied to the dataset for full number recognition the highest score obtained was 97.84% [6]. For whole digit recognition the aim of this project will be to exceed the performance of DNNs when applied to this dataset so any increase on 97.84% will be a success. Since the benchmark human accuracy is at 98% ithe aim will be to not be more successful that DNN recognition, but to exceed the 98% accuracy of human viewers. A secondary evalution metric for the success of this project will be in the time it takes to analyse the whole data set. When DNNs were applied to the data it is stated that training the model took 6 days. If a smaller time frame can be achieved this will add to the success of the project. I will also state that if a lower score than 98% accuracy is given by the end of the project, but the training time for the algorithm is sufficiently short then the project can be considered a success as long as the accuracy is still higher than the 90.6% found in the initial Stanford study.

Project Design

The theoretical workflow will start with data preparation. The cropped image portion of the dataset has already been normalised to 32x32 dimensions and will need little data preparation. The full number images will however have to be normalised to standard size, so more time will need to be dedicated to properly preparing this portion of the data.. After preparation the first stage of the algorithm will have a learning model learn to recognise single digits from the pre prepared dataset of 32x32 images. Once the algorithm has reached a successful accuracy the best weights will be saved and used as the basis for the next stage of the algorithm. The trained single digit algorithm will next be applied to the full images containing all digits in the address number. 32X32 sections of the full cropped images will be scanned, the 32x32 section will then step a set amount of pixels in one direction and the next section will be scanned. The likenesses of each number being contained in each segment of the image will be stored in an array as they are found. When the array is completed it will form the input to a second learning model. The sedon model will then be trained to recognise patterns and output the most likely address number for each image between 0 and 999.

Figure 1: The overall workflow of the algorithm

As mentioned above some data preprocessing will be required. Firstly the full numbers for each image will have to be collated as currently the data is separated only into single digits. Fortunately the single digits that have been manually colected and the image they are associated with is stored as a string. It should be relatively simple to create a dictionary containing the image name as a key and the address number as the value. The next stage of preprocessing the data is in normalising the image data for each portion of the dataset. The first set of data contains 32x32 images of each single digit this was collected specifically to train a single digit recogniser. This data will require little to no preprocessing, the only change will be if I convert the images to greyscale. The second set of images will require more processing as they are in variety of sizes and shapes. A specific size that is dividable by the step size of the algorithm and the 32x32 scan shape will have to be decided upon and all the input images normalised to this size. Again both colour and greyscale will be tested for performance.

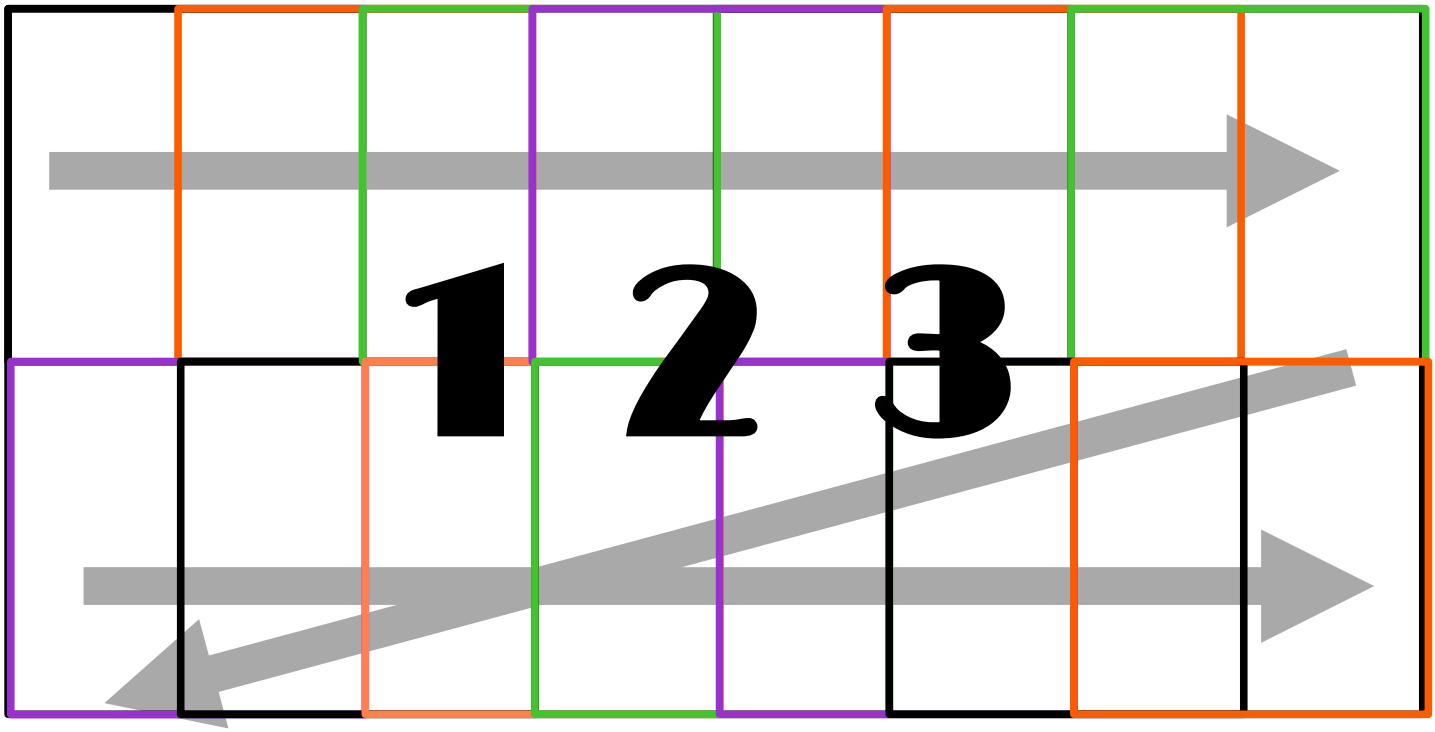

Figure 2: Cropped and overlapping portions of the full image

Figure 2 above shows how the full images may be separated into smaller overlapping windows of image data. The algorithm will scan each window separately and output the likeliness of a digit occurring in that window. In Figure 2 for example, we may see low likinesses for all numbers for the first 2 windows but a high percentage for 1 occuring in the third window with a smaller percentage for a 2 having occurred, The input to the second learning model will be an array containing the stored the likeliness of each number occurring in each window. The second learning model will recognise the regularity of digits appearing and will translate this to full 3 digit numbers. Deep Neural Networks (DNNs) could be used for the seond model since they tend to be more accurate. DNNs also take a long period of time to train. Conversely smaller neural networks take much less time to train but often lack the accuracy of a DNN.

A point of contention for the design of the second learning model will be in the number of outputs this model will have. Since with 3 digit numbers there are up to 1000 potential outputs (0-999) and in this case we should never have an address of 0 designing the number of outputs could cause problems. The output designs that will be tested first will be to have either 1000 or 999 outputs where the unit output with the highest value will represent a number between 0-999 or 1-999. Th other option would be to have 10 outputs and treat the output number as a binary number. The outputs from the net would be forced to either 1 or 0 depending on a tested threshold and the output number would be calculated from there. The problem here is that 10 digits is the minimum number of digits needed to represent 999 in binary however, the highest decimal number that can be achieved with 10 binary digits in 1023. 1023 is larger than our highest possible number and adding 24 more possible outcomes may reduce the accuracy of the algorithm. All approaches detailed above will be tested and the best course of action taken.

Below is a portion of pseudo code outlining the top level of the algorithm

```
import Data_Full_images_Train,  Data_Full_images_Test
import Data_32x32_Train,   Data_32x32_Test

# Preprocess Data
Processed_Full_Train = Crop_and_Greyscale(Data_Full_images_Train)
Processed_Full_Test = Crop_and_Greyscale(Data_Full_images_Test)
Processed_32x32_Train = Greyscale(Data_32x32_Train)
Processed_32x32_Test = Greyscale(Data_32x32_Test)

# Train First Model
First_Model_Weights = Single_Digit_Recognition(Processed_32x32)

# Scan through every image
for image in Processed_Full :
      # Scan Full Images and output likeliness to array
      Digits_Occurring_Array = Scan_Full_Images(image, First_Model_Weights)

      # Train Full number recognition model with weights
      Full_Number_Recognition(Digits_Occurring_Array)

# Test Full Number Recognition
Final_Algorithm_Accuracy =  Full_Number_Recognition_Final(Processed_Full_Test) /
len(Processed_Full_Test)

print(Final_Algorithm_Accuracy)
```

Overall I believe this will be an effective method of teaching an algorithm to recognise digits in real world images. The approach takes inspiration from human interaction and machine learning studies and combines the two approaches. I am confident that I will be able to design a machine learning algorithm that improves upon previous research and could be used in future commercial applications.

References

[1] Le Cun, Y., Boser, B., Denker, S., Henderson, D., Howard, R., Hubbard, W. and Jackel, D. (1990). Handwritten Digit Recognition with a Back-Propagation Network. *Advances in Neural Information Processing Systems*, 2, 396–404.

[2] Cireşan, D., Meier, U., Gambardella, L. and Schmidhuber, J. (2010). Deep, Big, Simple Neural Nets for Handwritten Digit Recognition. *Neural Computation*, volume 22, p.3207-3220.

[3] Cireşan, D., Meier, and Schmidhuber, J. (2012). Multi-column Deep Neural Networks for Image Classification. *Technical Report No. IDSIA-04-12*, IDSIA / USI-SUPSI.

[4] http://ufldl.stanford.edu. (2011). The Street View House Numbers (SVHN) Dataset. Available at: http://ufldl.stanford.edu/housenumbers/

[5] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, Andrew Y. Ng. (2011). Reading Digits in Natural Images with Unsupervised Feature Learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*. Available at: http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf

[6] Goodfellow, I., Bulatov, Y., Ibrarz, J., Arnoud, S. and Shet, V. (2014). Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks. *International Conference on Learning Representations*.