Identifying hand-written text, Canadian postal codes using Artificial Neural Network

Lavish Handa 110006198 Handal@uwindsor.ca

Karan Singla 105172747 singlak@uwindsor.ca

Abstract

Technology has been booming with every iteration and the field of artificial intelligence is so huge that it is still evolving for more than 50 years. With new data available on the internet every now and then expectations also rise and with expectations come new products which can work in fast paced environments. Artificial neural networks are one of the software products of Artificial Intelligence which can work in fast paced environment and can solve huge business problems which a lot of companies have to deal with the human workforce. With this product we would like to give a solution to different businesses who deal with a lot of characters and numbers on a daily basis and require human. Those characters or numbers can be full sentences or words or a combination of both to form something meaningful. A software like this can be used to solve problems like postal code in Canada where human readable format can be predicted by a software. Human efforts to navigate the incoming and outgoing posts (within Canada) can be reduced and given to this software. Like the postal code industry example, transaction number, IFSC code of banks written by humans while depositing a cheque can also be a problem this software can solve. Although the prime condition for this software is accuracy, this software can give extremely high accuracy for just the digits but lacks somewhere when looking at the characters.

INTRODUCTION

Overview

Since the time of Internet new technologies have been evolving and impacting different businesses around the globe. Solving business problems using technology is the way to leap forward and so is this project supposed to do. If we are able to produce machines which can take up the job predicting characters and numbers, a lot of human effort can be reduced albeit the product should be accurate enough add value and not problems. The problem we are looking at is the problem of predicting characters from a hand-written Canadian postal code. A postal code predictor can be useful to the postal company and humans working in the warehouse to navigate the posts according their postal numbers.

Motivation

The motivation behind this project is the people working in postal companies who sometimes note down the wrong information and the post may sometimes be sent to the wrong location or when people get the status which says ***"product lost in transit"*** is

common in such a large country. Postal codes can be confusing and can also add up to the time a person has to wait for the package.

Significance

This project's significance will be to reduce the load on the humans navigating parcels based on their code for specific provinces and fasten the process of navigating the posts. The more significant result will be to increase the throughput of the postal companies while navigating and reduce the lost transits and increasing human efficiency.

Contribution

Dataset study- Lavish

Defining neural network architecture- Karan

Building model- Lavish and Karan

Saving, Loading model- Lavish

User Image Preprocessing- Karan

Image Segmentation and Prediction- Karan

LITERATURE REVIEW

In the last decade, so many advancements in the tech world has given a new shape to the social media platform. Several companies are now using interactive platforms to help their customers save time. The patients expect faster responses from healthcare service providers. Delay in response time only annoys the customers and thus, to deliver the best services in time, firms are embracing technology trends such as embedding a product predictor to their website to stay at par with the latest trends. Such sort of automation in the fashion industry has reduced the load on industries by manifold. Professionals can now better focus on valuable things while the predictors assist customers with their queries, guidance, small operations and other matters. For their product-related issues, patients are turning to online predictors first as they have a quick response time and higher availability. Soon, a time will come when customer's first choice will be product predictors for help. Product/Words do have huge potential for success in different industries. In the recent past few years, the number of Artificial intelligence technologies has seen a growth on graph.

PROJECT DETAILS AND OVERVIEW

This section of the technical report presents the project details and the methodology that has been adopted for the development of this project.

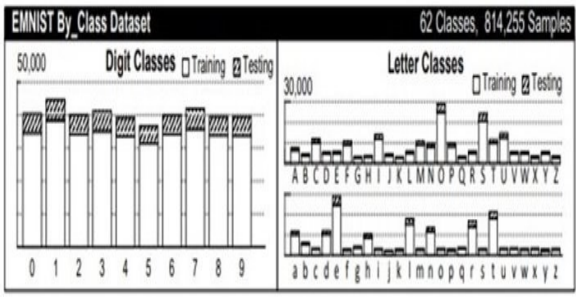
Definitions

Major definitions essential for the appropriate understanding of this project are mentioned below: -

- **Artificial intelligence (AI)**: Artificial intelligence (AI), the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings. The term is frequently applied to the project of developing systems endowed with the intellectual processes characteristic of humans, such as the ability to reason, discover meaning, generalize, or learn from experience.
- **Artificial Neural Networks (ANN)**: An artificial neuron network (ANN) is a computational model based on the structure and functions of biological neural networks. Information that flows through the network affects the structure of the ANN because a neural network changes - or learns, in a sense - based on that input and output.
- **TensorFlow**: Created by the Google Brain team, TensorFlow is an open

source library for numerical computation and large-scale machine learning. TensorFlow bundles together a slew of machine learning and deep learning (aka neural networking) models and algorithms and makes them useful.

- **EMNIST Dataset**: The EMNIST dataset is a set of handwritten character digits derived from the NIST Special Database 19 and converted to a 28x28 pixel image format and dataset structure that directly matches the MNIST dataset.
- The ByClass Dataset



ALGORITHM

Following steps were followed to implement the characters recognition for Canadian Postal codes:

1) Dividing the EMNIST "byclass" dataset into training and testing sets.

2) Normalizing the training and test data.

3) Defining the artificial neural network architecture by adding multiple layers and creating the model.

4) Building the neural network model.

5) Adding the optimizer function "ADAM" to reduce the loss error.

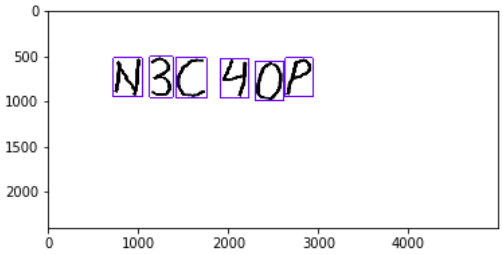
6) Fitting the model with training data and then testing it on test data from EMNIST.

7) Calculating the accuracy, saving the model.

8) Loading the real-world image with handwritten characters

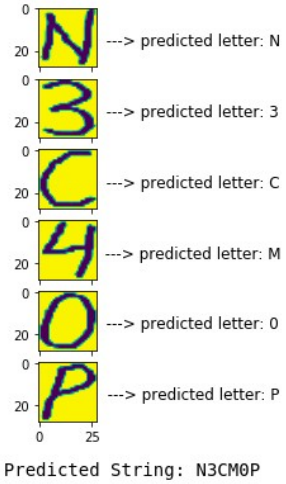
9) Preprocessing the image, converting into 28*28 pixels with EMNIST dataset format.

10) Segmentation of the image and identification of each character present in the image.



11) Finally, checking the modified image on the model and obtaining the predicted string.

12) The predicted string can be used and passed through any api to detect the canadian postal code.



EXPERIMENTAL SETUP

A) Implementation Details

1) Requirement analysis. Import necessary libraries like TensorFlow to use the functions, EMNIST dataset to run the operations on and matplotlib for plotting our images and results.

2) Using the byClass dataset which is the most reliable dataset available in the various datasets available in EMNIST to extract the already present training and testing samples.

3) Start the training process with the ***ANN*** implementing the number of layers in the input, hidden and output layers.

4) To improve the accuracy, we must choose a number of iterations for back-propagation so that the weights and biases can be changed and we can get improved result which is used in the Tensorflow as **"epoch"**.

5) User image has to be preprocessed, background color is changed and pixels are converted according to EMNIST dataset 28*28 pixels.

6) After preprocessing of the image, segmentation is done according to the normalized vectors in the image and a rectangle is drawn around the characters using OpenCV.

7) Finally, the new modified image with segmentation is passed into the model and the prediction is observed.

B) Challenges

1) One of the major challenges we faced in our project was when we used our own self created image as a testing dataset, the accuracy dropped. The accuracy was good when we were testing on the dataset test samples. But when real time image is used, image has to be preprocessed according to the model trained.

2) Segmentation of the image since the number of characters can vary. Recognizing each and every character in the image was very difficult.

3) Increasing the number of epochs doesn't necessarily mean that the accuracy is going to increase. Overfitting can be an issue. Depending upon the discrepancies in the image, even more iterations would not increase accuracy.

C) Limitations

1) Some alphabets are written the same way in uppercase and lowercase both with only difference being size. Though for the postal code it won't be a problem but if case of an alphabet is a concern then this may not be considered a problem solver.

2) If an image is not clear and some pixels do show something which is not meaningful to the human eye, the predicted result will show it most of times as the number zero which is not the case. The non-meaningful pixel on the image can be an innocent mistake as well

FINDINGS

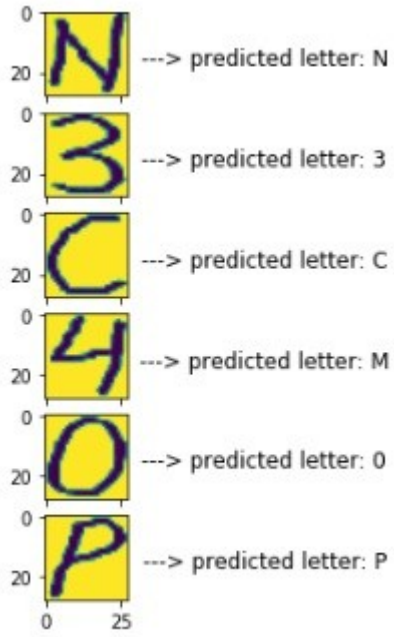
1) Increasing the number of iterations for back-propagation would not definitely lead to better results. Even if we do get better results, the difference would be negligible. Example- In the 20^{th} iteration the accuracy dropped in comparison to 19^{th} iteration

```
Epoch 1/20
697932/697932 [==============================] - 57s 82us/sample - loss: 0.6283 - accuracy: 0.794
Epoch 2/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4846 - accuracy: 0.831
Epoch 3/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4573 - accuracy: 0.839
Epoch 4/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4424 - accuracy: 0.843
Epoch 5/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4317 - accuracy: 0.846
Epoch 6/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4248 - accuracy: 0.847
Epoch 7/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4188 - accuracy: 0.848
Epoch 8/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4149 - accuracy: 0.850
Epoch 9/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4121 - accuracy: 0.851
Epoch 10/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4084 - accuracy: 0.852
Epoch 11/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4066 - accuracy: 0.852
Epoch 12/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4047 - accuracy: 0.853
Epoch 13/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4032 - accuracy: 0.854
Epoch 14/20
697932/697932 [==============================] - 56s 81us/sample - loss: 0.4020 - accuracy: 0.854
Epoch 15/20
697932/697932 [==============================] - 57s 82us/sample - loss: 0.4005 - accuracy: 0.854
Epoch 16/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.4000 - accuracy: 0.854
Epoch 17/20
697932/697932 [==============================] - 56s 80us/sample - loss: 0.3994 - accuracy: 0.855
Epoch 18/20
697932/697932 [==============================] - 55s 79us/sample - loss: 0.3987 - accuracy: 0.855
Epoch 19/20
697932/697932 [==============================] - 55s 79us/sample - loss: 0.3975 - accuracy: 0.855
Epoch 20/20
697932/697932 [==============================] - 55s 79us/sample - loss: 0.3976 - accuracy: 0.855
<tensorflow.python.keras.callbacks.History at 0x7fd9340661d0>
```

2) The algorithm will give some form of result whatsoever. If an image has something unreadable the algorithm will most probably return it as a 0(zero). Some images which had the alphanumeric as well as something unreadable on any side, the algorithm returned it with 0(zero).

3) Our accuracy dropped significantly when we included characters with numbers for the Canadian Postal Code. Previously when we were using just digits, we were constantly hitting 98-99% accuracy but with addition of characters we've even seen an accuracy as low as 84%.

4) Similar looking characters are often the ones which are found to be different in the output and have the maximum chance of being interpreted something else. Example- In this case the number "4" and the letter "M" have somewhat similar build and size.



CONCLUSION

This tool will help in reducing human effort of the Canadian postal code. Since the Canadian postal code is independent if the case of the alphabets used for navigating the mails, a software like this can benefit the postal companies in navigating the posts. Therefore, human stress on the posts can be reduced significantly and this software can be a huge game changer in the industry. Just like this word predictor there is a significantly widely used technology what we call chatbots which work on ANN and help businesses add value with an increase in the efficiency. Chatbots are also made to be intelligent using ANN and give more accurate results.

Future Scope

• New features can be added to improve the accuracy of the recognition where accuracy dropped the most.

• This algorithm can be used on a large database of handwritten English text like a paragraph.

• The proposed work can be extended to work on degraded text or broken characters.

• Recognition of digits in the text, compound characters and half characters can be done to improve the word recognition rate which in turn will elevate accuracy.

• Using the algorithm, an OCR based application can be developed.

ACKNOWLEDGEMENT

We would like to express our gratitude towards our course instructor Dr. Dan Wu, who motivated us on every phase and convinced us to try new directions, challenges as well. Without his tenacious guidance and motivation to do something new, this project would not have been what it is today.

REFERENCES

E.Birbek. "How to Build a Neural Network to Recognize Handwritten Digits with TensorFlow" Internet:"https://www.digitalocean.com/community/tutorials/how-to-build-a-neural-network-to-recognize-handwritten-digits-with-tensorflow" Mar 28, 2019 [December 20, 2019]

H.Scheidl. "Build a Handwritten Text Recognition System using TensorFlow" Internet: "https://towardsdatascience.com/build-a-handwritten-text-recognition-system-using-tensorflow-2326a3487cd5 " Jun 15, 2018 [December 20, 2019]

P.Dwivedi. "Handwriting recognition using TensorFlow and Keras" Internet: "https://towardsdatascience.com/handwriting-recognition-using-tensorflow-and-keras-819b36148fe5" Jan 21, 2018 [December 2, 2019]

Agarwal M.; Shalika; Tomar V; Gupta P. "Handwritten Character Recognition using Neural Network and Tensor Flow" International Journal of Innovative Technology and Exploring Engineering (IJITEE) Volume-8, Issue- 6S4, April 2019

Computer Science. "Classify Images Using Python & Machine Learning" Internet: "https://www.youtube.com/watch?v=mB7fdy67eFw" Jul 11, 2019 [November 28, 2019]