Synopsis

on

Project title

Submitted in the partial fulfilment of the Degree of Bachelor of Technology (Computer Science and Engineering)

Submitted by

Name of group members with enrollment no

Under the supervision of

Name of mentor

Department of computer Science and Engineering

Maharaja Surajmal Institute of Technology
Janakpuri, New Delhi.
2018-22

INDEX

Abstract

In this project we seek to digitise and evaluate online (as the user is writing) and offline (from static image) handwritten mathematical expressions. For this, we will be using OpenCV for image processing and Tensorflow for training a Convolutional Neural Network (CNN) for recognising the operators, operands and letters in the mathematical equation. After the mathematical equation has been digitised, it will be solved using our algorithm written in Python.

Introduction

While a large number of researches have been done for digits and character recognition, much less progress has been made surrounding handwritten mathematical equation recognition. While typing is much faster than writing by hand, mathematical equations have the opposite property: writing mathematical equations by hand is more efficient than typesetting them.

Furthermore, whereas handwritten equations are human-readable, the "code" of typesetting languages are often highly nested and difficult to edit. These problems of typesetting presents an opportunity for improving the workflow of writing mathematical equations digitally.

The aim of the project is to digitise the handwritten mathematical equation input by the user in the form of an image and then evaluate the equation to produce the correct answer.

But it is generally easy for a human to recognize handwritten characters and symbols, and difficult for a computer to recognize them. This difficulty can be overcome by first, segmenting the components of the equation and then using a deep learning based approach by designing a neural network that recognizes the characters based on a large trainable dataset of digits, letters and mathematical symbols.

Literature Survey

A lot of work has been done on recognizing handwritten numbers and English words. The MNIST dataset is a very popular dataset to try out machine learning models. But in our project we will be using two different datasets as we will require digits, letters and mathematical symbols to train out deep learning model:

1. The EMNIST dataset encompasses 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[8].

2. The "Handwritten math symbol dataset" available on Kaggle to recognise mathematical symbols is required for training the model to learn the mathematical symbols[2].

Earlier work in this field has entailed the use of more traditional ways to segment and recognize mathematical symbols. For example, computing centroid and using clustering to segment data, using geometric features for recognition, etc[7].

Due to the release and application of TensorFlow, papers after 2015 are more relevant. Most papers suggest methods of partitioning an expression into individual mathematical symbols, then use a trained convolutional neural network model to recognize individual symbols, and finally use an algorithm to reconstruct equations as suggested in [5].

For the architecture of CNN model, we refer to [6] to achieve the best accuracy possible for predicting the mathematical digits and symbols.

Objective

The main objective of our project is to digitise and evaluate handwritten mathematical expressions. We approach this problem:

Objectives Pointwise (Write objective of project in points)

1.
2.

Research Methodology

Our approach can be divided into 6 steps:

Dataset preparation

The dataset required for this project would require to have images of digits, operators and special characters such as "(" and ")". To do such we would combine the MNIST dataset and several other datasets, which will allow minimum data preprocessing. If any of the characters have fewer images than a threshold value, then the images will be augmented to make enough data points for the training of deep neural network model.

Image preprocessing

All the images present in the created dataset should be equal in size and to make sure that we convert all the images to a square image each having one channel.

Training deep neural network

A deep neural network model would be defined as having layers such as Convolutional layer, Maxpooling layer, Activation layer and Fully connected layers. When the model is compiled, it will be trained on the custom-created dataset.

Text segmentation

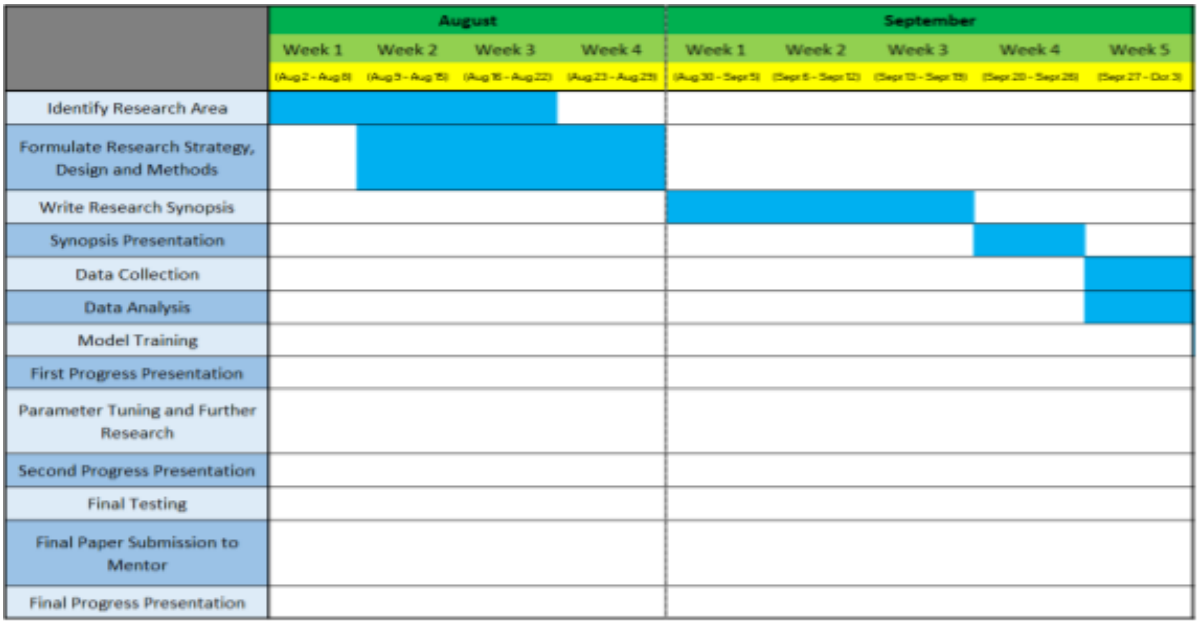
This step would extract the digits and operators from the input image. OpenCV would be used to make rectangular contours around individual characters and the content of each contour would be stored separately to be fed to the deep neural network.

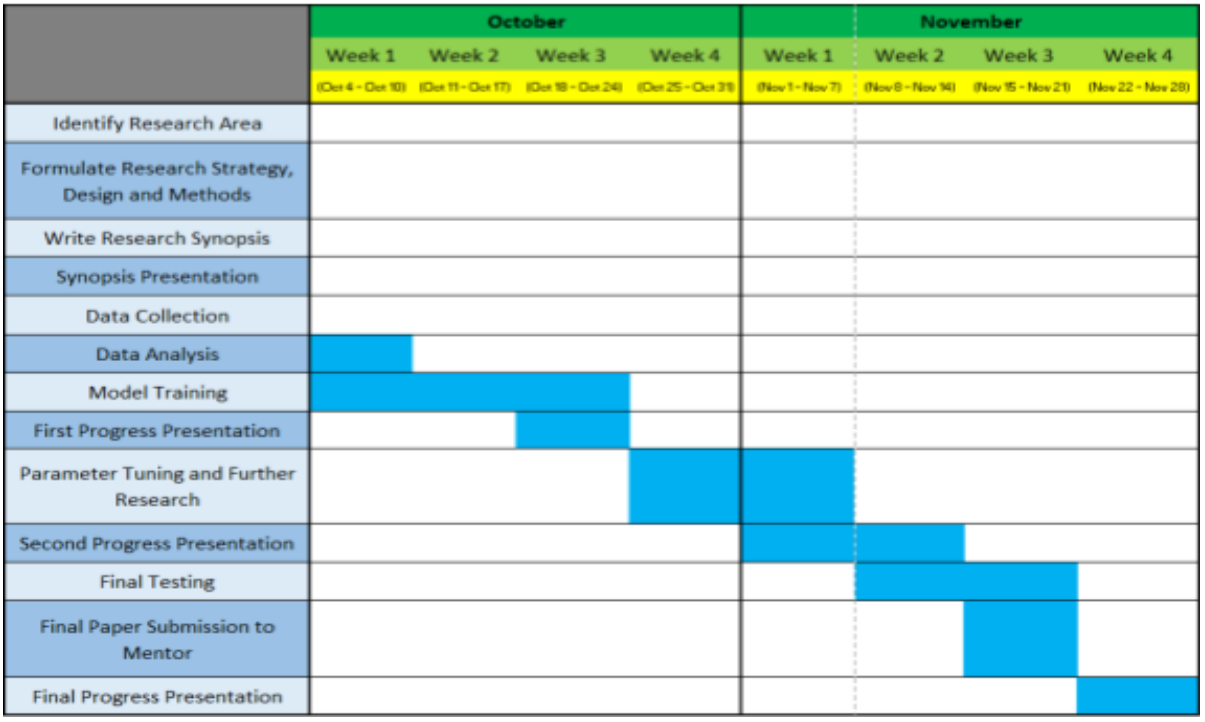
Each segmented image may have a different size, and these images can not be fed directly as an input to the deep neural network model. The images need to be resized. All the images would be converted to square-shaped images and having the size same as the size of our preprocessed images.

Each segmented image will contain a symbol that will act as an input to the deep learning model. The characters at each level would be passed as input from left to right, and this process will be repeated for every level from top to bottom. The images would be predicted by the model and the predicted label of each image would be appended to an empty string.

The string containing the expression would be converted into an actual expression that a computer can solve and the result of the expression would be generated.

Gantt chart





References

1. Agarwal, S., Rani, A., Singh, V. and Mittal, A.P.,*" EEG signal enhancement using cascaded S-Golay filter,"* Biomedical Signal Processing and Control, 36, pp.194-204, 2017.
2. A. Yadav and D. K. Vishwakarma, "*A Weighted Text Representation framework for Sentiment Analysis of Medical Drug Reviews*," IEEE Sixth International Conference on Multimedia Big Data (BigMM), pp. 326-332, 2020