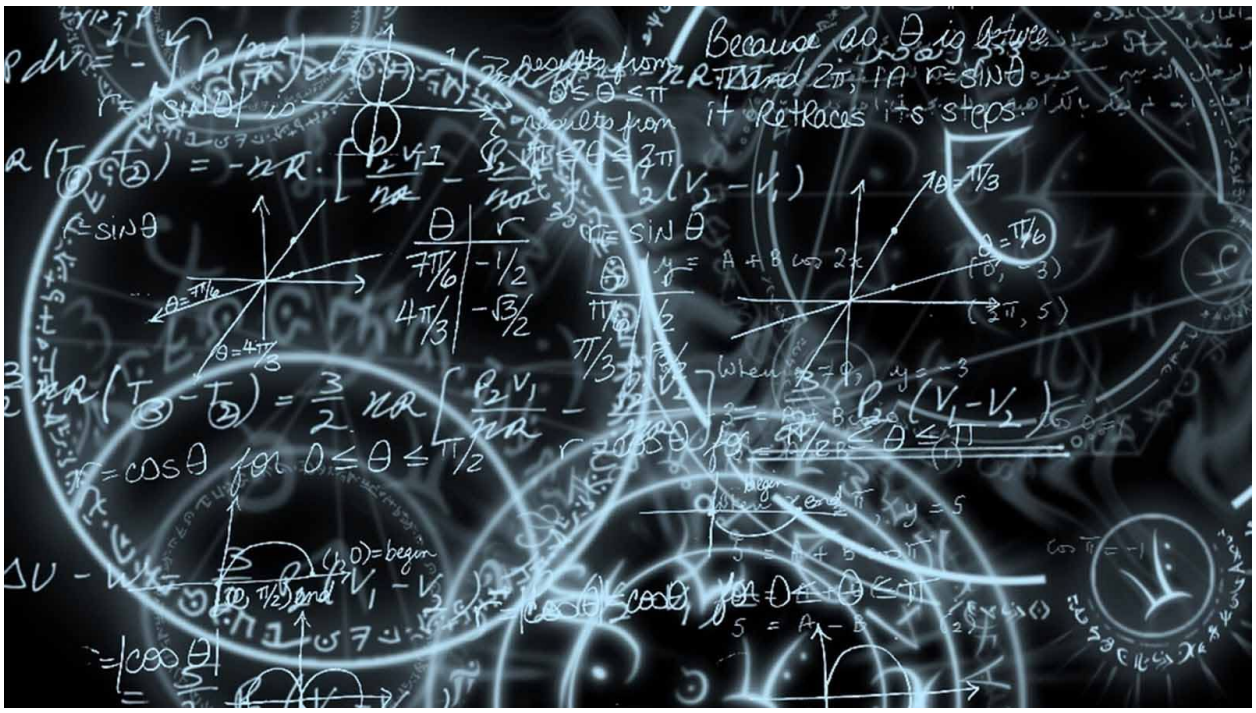

Android based Mathematical Expression Evaluation from Images

15.02.2018

Venkataraman S
15CO140

Kiran S
15CO125

Madhu M
15CO226

Overview

Optical character recognition (OCR) is an important technique for solving many technical problems. As a few examples, OCR has been used for digitizing books, identifying license plates, and assisting the vision impaired. In this report, we propose to develop an Android mobile application that uses OCR to recognize and evaluate arithmetic expressions. The app could be used to help students check their solutions to challenging problems quickly and easily.

Goals

1. For our initial scope, we plan to focus on recognizing digits (0-9) and basic mathematical operators (+, -, x, and /), and displaying the computed result on the device's viewfinder.
2. Finally, we will improve the app to recognize and evaluate integration and differential equations.

Implementation

The main stages of the proposed application development will include:

- **Keypoint Detection** - The first step of the application is to find text in the image. We first convert the image to grayscale and detect maximally stable extremal regions. MSER detection searches for regions of an image that remain relatively stable across many different thresholds. In other words, it finds regions that are darker than a certain value. As that value is increased, those regions will grow. Regions that do not grow while the value is significantly increased are those with much darker (or lighter) values than the region around them. MSERs are useful features for text detection because of the strong contrast between text and background and because of text's connectedness. This would reduce the number of spurious keypoints found

by MSER, resulting in better bounding boxes around equations in the next step of the pipeline.

- **Equation Bounding** -We construct bounding boxes around them. This provides us with a rectangle around the text. Depending on the scale of the image, each rectangle might bound an entire expression or, in the case of large font with large gaps in between, it might bound each character by itself.

- **Thresholding -** Removing shadows, noise, and other artifacts that could interfere with character recognition. To mitigate these artifacts, we perform locally adaptive thresholding over the entire image . Locally adaptive thresholding is particularly useful for this application, because parts of the captured image are often under different amounts of shadow (for example, when the phone casts a shadow over part of the image). Alternatively, one may choose to apply thresholding to the interior of each bounding box individually. However, one still may need to apply locally adaptive thresholding to each bounding box, as some boxes may have non-uniform illumination.

- **Character Recognition** - Characters and symbols will be recognised by OCR engine. The contents of each bounding box are then sent to the OCR engine. We use the Tesseract OCR engine, which is an open source project currently supported by Google. To increase accuracy, we set Tesseract to match the text only to the symbols of interest, i.e. 0-9 and “+-()/x”. Along with the symbols used in differentiation and integration.

- **Evaluation** - The OCR engine then sends the mathematical expression to be evaluated.

- **Display** - After obtaining the result of the expression, we overlay the result on the mobile device’s screen next to the corresponding equation. Note that we display the captured image on the device throughout steps A-E as well. Moreover, we color-code the bounding boxes to indicate progress. When bounding boxes are found, we display yellow rectangles around them, As the contents of each bounding box is parsed and evaluated, we change the

border color to green (if the evaluation is successful) or red (if evaluation is not successful). An overview of the Tesseract OCR engine. The result is either displayed on the top or bottom of the box.

Weekly Plan

I. Week - 1

Learning App development in Android Studio.

II. Week - 2

Getting through the basics of Machine Learning and Neural networking.

III. Week - 3

Studying various OCR algorithms and other algorithms related to the project.

IV. Week - 4

Implementing OCR algorithms and recognising texts from the images.

V. Week - 5

Improving the algorithm to recognise integral and other mathematical operators.

VI. Week - 6

Evaluating the basic arithmetic expressions.

VII. Week -7 and 8

Evaluating calculus expressions and other complex expressions.

VIII. Week - 9 and 10

Cleaning the code and wrapping it up.

Progress

As we proposed earlier we have built a basic version of the Android Application that evaluates simple mathematical expressions such as addition, subtraction, multiplication and division. So far the progress include Optical Character Recognition (OCR) of basic mathematical expressions, evaluations and displaying the results.

Added training dataset for Tesseract has been added which increases the accuracy of the OCR.

Future Work

Along with this we will implement integration and differentiation expression evaluation which will be shown in the End progress.