OCR FOR MATHEMATICAL EQUATIONS

1. **Overview:**

To develop deep learning algorithm solution package for recognizing and digitizing steps of solving a mathematical equation written by freehand on a paper.

1. **Dataset:**

Dataset need to be generated for a specific type of equation (Binomial Algebraic Equation). We can generate the mathematical expressions by creating equations of randomly selected numbers, variables and operators from handwritten mathematical symbols dataset from kaggle. Worksheet format (background) given can be added to the generated image.

* 1. **Solving mathematical equations pipeline:**

**Step 1:** Detect each line in the image and crop accordingly.

**Step 2:** Perform OCR on each detected line.

**Step 3:** Evaluate the equation by solving each line and provide image output with colored bounding boxes with right or wrong indication.

* 1. **Training, testing and validation data set:**

The generated dataset is split into training, testing and validation subsets in the ratio of 8:1:1

* 1. **Data Preprocessing:**

All the approaches works best on preprocessed images. We need to remove the noise, but the low frequency characters like “**.**”, “-” and “**,**” should be enhanced in such a way that it is not lost after multiple pooling layers.

1. **Approaches** 
   1. **Tesseract:**

Tesseract 4 adds a new neural net (LSTM) based OCR engine which is focused on line recognition. It also has a unique configuration option for detecting equation region in the document.

**Drawbacks in tesseract:**

* + - Works best on clean and computer generated images.
    - Incorrect detection of numbers.
    - Unable to recognize the exponents of the base.
    - Operators are recognized as numbers.
    - Unable to recognize decimal points.

We may need to retrain the tesseract on handwritten mathematical expressions to improve accuracy.

**Goals:**

* Split the generated datasets into train/ test dataset
* Run tesseract on the cropped equation line images and establish a baseline performance of tesseract without any retraining or fine-tuning
* Data files should be prepared in the format accepted by tesseract.
* Re-train or fine tune tesseract to recognize mathematical expressions.
* Evaluate the performance of the retrained/ fine-tuned tesseract on the test dataset.

**File Format accepted by tesseract:**

Images should be converted to .tif format, Tesseract needs a 'box' file to go with each training image. The box file is a text file that lists the characters in the training image, in order, one per line, with the coordinates of the bounding box around the image, we can use OCR-D (a tesseract 4 wrapper) for that.

*\*Not sure how to represent exponents in the image*.

* 1. **Off-Line Math Formula Recognition:**

This model is based on the paper [Multi-Scale Attention with Dense Encoder for Handwritten Mathematical Expression Recognition](https://arxiv.org/pdf/1801.03530.pdf) uses attention based encoder-decoder. This model has been trained on handwritten mathematical expressions. To retrain this model we need GPU and the images need to be normalized to get better results. Tesseract fails in recognizing characters with low frequency but this approach scores high in such scenarios by adding multi-scale attention decoder with Dense Encoders.

* 1. [**Image-to-Markup Generation with Coarse-to-Fine Attention**](https://arxiv.org/pdf/1609.04938.pdf)**:**

This approach also presents attention based encoder-decoder model to convert images to LaTex. It majorly revolves around rendered markup images and less focus was given to hand written mathematical expressions, but this approach doesn’t require heavy image preprocessing.

One of our solution for the given objective is to ensemble the above 3 approaches and rank individual approach to get the optimal result.