Proposal for the project: system of linear equations solver with deep learning

Problem

Over the course of the last months e-learning gets more and more to the focus of the broad population. The Covid-19 pandemic forces nearly every country in the world to apply restrictions on the number of people allowed to gather in public spaces including schools. One of the major effects of these lockdowns is the breakdown of public education as we know it. Students, teachers and parents have to turn to modern and innovative approaches for teaching on a remote basis.

This trend results in a rising demand for digital solutions in the education sector enabling students to learn at home and at their individual pace. Furthermore these solutions enable children in less developed countries to catch up with their peers in the so-called First and Second World without having access to a widespread and sophisticated public education infrastructure.

Mathematics is one of the essential domains in the course of every child's formal education providing the necessary skills to thrive in the modern technical and digitalized world. Systems of linear equations are a central subject of every Mathematics curriculum. Enabling to solve problems ranging from flows in complex systems to analytical geometry.

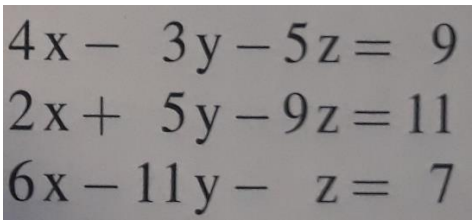
This project tries to provide a tool for solving these systems with the help of a image recognition deep learning model. Users should be able to upload an image via a client (browser) and get the result of the systems of equations back to their device in near real-time.

Data

As a first approach we restrict these equations to the common mathematical variable identifiers x, y and z as well as integer numbers consisting of the literals 0, 1, …9 as operands and -, + and = as operators.

Our model has to be trained on a set of images representing the set of these symbols. This project relies on part of the dataset provided by Xai Nano on Kaggle consisting of images for single mathematical symbols (https://www.kaggle.com/xainano/handwrittenmathsymbols).

For validation of the model the user-uploaded images of the systems of linear equations are needed and have to split into a series of the above mentioned symbols. This set of image has to be created manually by taking pictures from Maths books.



Solution

The solution to the given problem consists of two major parts.

The first part handles the user image upload via a simple web page enabling the client to send data via an API to a remote server. There the images will be preprocessed and split into lines and images of single mathematical symbols.

The deep learning model forms the basis of the second part. It has to be trained on a set of preprocessed images of single mathematical symbols and evaluated on a holdout set. It will then be fed with the preprocessed user-uploaded image parts and outputs a digital representation of the equations that can be evaluated. The solution can then be transferred back to the user's client.

Model evaluation metrics stated for several models on the official webpage of the MNIST handwritten digits dataset provided by LeCun et al. (http://yann.lecun.com/exdb/mnist/) serve as a benchmark for the single symbol image classifier.

The user-uploaded image segmentation lacks an appropriate benchmark but can be evaluated based on a common metric e.g. accuracy of correctly segmented symbols.

Design

Given the two-part structure of the project the first focus lies on building the web architecture to enable image upload and transfer to the image classifier. Therefore a simple web page with an image upload has to be hosted or provided as a local HTML file. On the server side an API has to be designed allowing to provide images in a common format. These images have to be preprocessed and transformed to images showing only a single symbol each and fed to the model hosted on the server e.g. with the help of an Amazon AWS lambda function.

The model itself has to be trained and evaluated on a subset of the Kaggle mathematical symbols image dataset. The training will take place locally due to the massive dimensions of the dataset (up to 34,000 images for a single symbol). The model parameters will then be used to establish a model endpoint using Amazon Sagemaker that can be fed with the preprocessed data from the lambda function.

After the image classification the resulting equations will be evaluated with the help of inverse matrices and the solution send back to the web page on the users client.