CS230

Automated LaTeX Code Generation from Handwritten Mathematical Expressions Category: Computer Vision

Jayaprakash Sundararaj
osjp@stanford.edu

Akhil Vyas
avyas21@stanford.edu

Benjamin Gonzalez-Maldonado
bengm@stanford.edu

Abstract

Training a model that learns handwritten mathematical expressions from images and generates equivalent LaTeX code. The goal is experiment and study different model architectures (CNN, LSTM, etc) and hyper-parameters, evaluate the with different evaluation metrics, and share our finding.

1 Introduction

Converting handwritten mathematical expressions into digital formats is time consuming, specifically LaTeX code. Our goal is to train a ML model that is capable of encoding handwritten notes and converting to the source code seamlessly. The input to our algorithm is an image of a handwritten mathematical expression. The challenge of our project is to convert an image to a text LaTeX sequence which will require the use of both computer vision and NLP techniques. We will use concepts related to these areas that we learn from this course to train the model. We will explore different evaluation metrics (text based, and image based), and share our findings.

2 Related work

Schechter et al. [2017] investigated a variety of methods like neural networks, CNNs, Random Forests, SVMs, OCR, CGrp, and SA. However, most state of the art the methods utilize encoder-decoder architectures involving CNNs and LSTM architectures like Genthial and Sauvestre [2017a]. In recent works like Bian et al. [2022], both left-to-right and right-to-left decoders are utilized. In our work, we will explore different hyper-parameters and model architectures such as attention mechanisms which were never tried before.

3 Dataset and Features

We will use the datasets from two main repositories: `Im2latex-100k` (Kanervisto [2016]) and `Im2latex-230k` (Gervais et al. [2024]). These datasets consist of images of mathematical formulas paired with their corresponding LaTeX code (two features). The Latex code is variable length.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

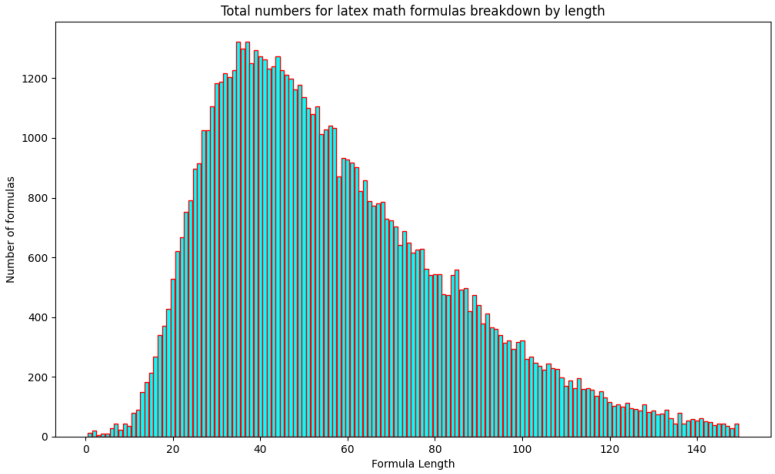


Figure 1: Formulas breakdown by length

The `Im2latex-100k` (Kanervisto [2016]) dataset, available at Zenodo, contains 100,000 image-formula pairs. Each image is in PNG format with a fixed size, and the formulas are extracted from ArXiv papers. This dataset is a cleaned-up version from the Cornell KDD competition (KDD Cup 2003). The `Im2latex-230k` (Gervais et al. [2024]) dataset, also known as `Im2latexv2`, contains 230,000 samples. It includes both OpenAI-generated and handwritten examples, further enhancing the diversity of the data. This dataset is available at Im2markup. The training data format is `<image file name> <formula id>`.

The dataset disk size is 849 MB. The images are gray scales with 50x200 pixels. The numbers of symbols (Figure 1) in the latex formulas vary from range varies from 1 to 150 symbols. Voabulary contains 540 symbols, refer Figure 4 and Figure 5 for the list of popular and least occurring symbols with their frequency.

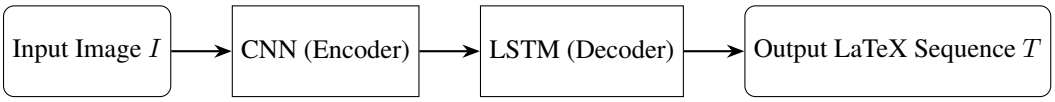
4 Experiment

In this project, we will use a Convolutional Neural Network (CNN) (O'Shea and Nash [2015]) combined with sequence-to-sequence (Seq2Seq) (Hochreiter and Schmidhuber [1997]) model to convert handwritten mathematical expressions into LaTeX code. The input to our algorithm is an image I, and the output is the corresponding LaTeX code, denoted as a sequence of tokens $T = \{t_1, t_2, \ldots, t_n\}$.

The Seq2Seq model, which consists of an encoder (the CNN) and a decoder (RNN, or LSTM), is trained for generate the LaTeX code token by token. At each time step t, the decoder predicts the next token t_t given the previous tokens and the context vector c_t. The probability distribution for the next token is computed as:

$$P(t_t|t_{1:t-1}, c_t) = \text{softmax}(W_o h_t)$$

where h_t is the hidden state of the decoder at time step t, and W_o is a weight matrix.



We'll use the cross-entropy loss function to optimize the model during training. Also, we plan to explore different attention mechanisms and extracting salient features as we iterate on the experimentation.

4.1 Experiment Seup

We use the single AWS P2 instance (tesla 80 GPU) for training. The training time varies between 1 hr 30 mins and 2 hrs. We use the ‘sparse categorical loss‘ with ‘adam‘ optimizer for 20 epoches. These hyperparameter are not modified between different configurations to observe the differences in outcome.

4.2 CNN encoder and GRU/LSTM

As a baseline, We use the CNN Encoder to encode the image input of resized image (50x200) with 1 channel (greyscale). We use 3x3 convolutional filter followed by 2x2 max pooling layer. This previous block is repeated three times and followed fully connected layer.

During decoding, We compute the embedding for formula tokens and concatenated with image encoded embedding. The concatenation of image and token embedding fed into LSTM units, followed by fully connected network. The activation is softmax. the LSTM instead of the GRU to compare the LSTM's performance improvement if any.

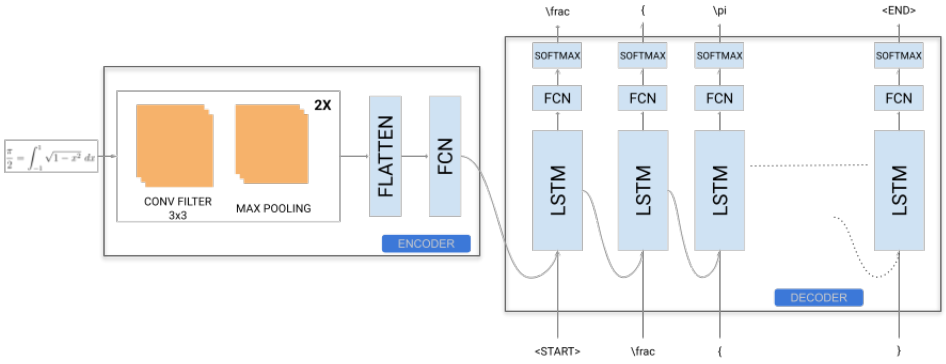


Figure 2: Baseline: CNN Encoder with LSTM Decoder.

4.3 LSTM with funetuning with pretrained Resnet50

In this experiment, we use the pretained ResNet50 model as a encoder (98Mb disk size). However, ResNet50 expects the image with fixed size 254x254 and 3 channels. Our input images are grey scale. So, we transform the input image to the ResNet50 input using `tf.keras.layers.Lambda(lambda x: tf.image.grayscale_to_rgb(x)`.

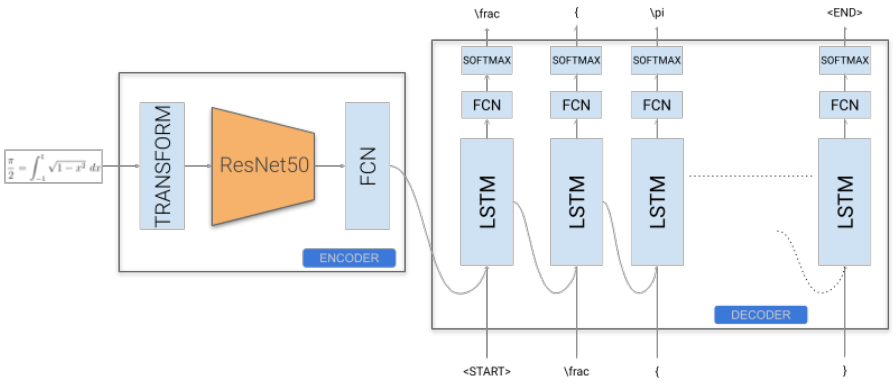


Figure 3: Pretrained ResNet50 Encoder with LSTM Decoder.

4.4 Re-weighted mis-predicted training example

We're aiming to inspect the accuracy losses and ensure that mis-predicted examples are correctly identified with increased weighting.

4.5 Attention Model

Work in Progress by Akhil. We're in the middle of using attention mechanism to experiment with attention mechanism for the model to learn complex dependencies.

4.6 Beam Search Instead of Greedy Decoding

Work in Progress by JP During inference time, we're using the greedy algorithm to pick the token with maximum logit score at every time. It terminates when the `<END>` token is predicted or reaches the maximum sequence length. We will explore beam search in this experiment.

5 Experiments and Evaluation

We will use the following metrics to evaluate the performance of the model for the LaTeX code generation task. The text based metrics compare the original LaTeX code with generated LaTeX code, however, the image based metrics compare the PDF images generated from original and generated LaTeX code.

- Text Metrics: **BLEU Score** (Papineni et al. [2002])
- Text Metrics: **Levenshtein Distance**
- Image metrics: Compute the **accuracy** between images generated from original latex code and generated latex code. Work in-progress by Ben

The experimentation will involve choosing different CNN layers (striding, pooling), learning rate, mini batch size.

6 Remaining work for Final Project

7 Contributions

Jayaprakash Sundararaj: Initial report, researching the dataset and existing methods. Implementing the full CNN and LSTM as a baseline. Extending to pre-trained ResNet50 model with finetuning.

Akhil: Ideation, Setting AWS/GPU setup, Extending to full CNN and GRU as a baseline, Extending to attention/transformer architecture (work in progress).

Ben: TODO

References

Amit Schechter, Norah Borus, and William Bakst. Converting handwritten mathematical expressions into latex, 2017. URL `https://cs229.stanford.edu/proj2017/final-reports/5241761.pdf`.

Guillaume Genthial and Romain Sauvestre. Image to latex, 2017a. URL `https://cs231n.stanford.edu/reports/2017/pdfs/815.pdf`.

Xiaohang Bian, Bo Qin, Xiaozhe Xin, Jianwu Li, Xuefeng Su, and Yanfeng Wang. Handwritten mathematical expression recognition via attention aggregation based bi-directional mutual learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 113–121, 2022.

Anssi Kanervisto. im2latex-100k , arxiv:1609.04938, July 2016. URL `https://doi.org/10.5281/zenodo.56198`.

Philippe Gervais, Asya Fadeeva, and Andrii Maksai. Mathwriting: A dataset for handwritten mathematical expression recognition, 2024. URL https://arxiv.org/abs/2404.10690.

Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *CoRR*, abs/1511.08458, 2015. URL http://arxiv.org/abs/1511.08458.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin, editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040.

Yuntian Deng, Anssi Kanervisto, and Alexander M. Rush. What you get is what you see: A visual markup decompiler. *CoRR*, abs/1609.04938, 2016. URL http://arxiv.org/abs/1609.04938.

Zelun Wang and Jyh-Charn Liu. Translating mathematical formula images to latex sequences using deep neural networks with sequence-level training. *CoRR*, abs/1908.11415, 2019. URL http://arxiv.org/abs/1908.11415.

Zelun Wang and Jyh-Charn Liu. Image to latex: A neural network approach, 2018. URL https://cs230.stanford.edu/projects_spring_2018/reports/8287695.pdf.

Guillaume Genthial and Romain Sauvestre. Image to latex, 2017b. URL https://cs231n.stanford.edu/reports/2017/pdfs/815.pdf.

Hongyu Wang and Guangcun Shan. Recognizing handwritten mathematical expressions as latex sequences using a multiscale robust neural network. *CoRR*, abs/2003.00817, 2020. URL https://arxiv.org/abs/2003.00817.

Daniil Gurgurov and Aleksey Morshnev. Image-to-latex converter for mathematical formulas and text, 2024. URL https://arxiv.org/abs/2408.04015.

8 Appendix

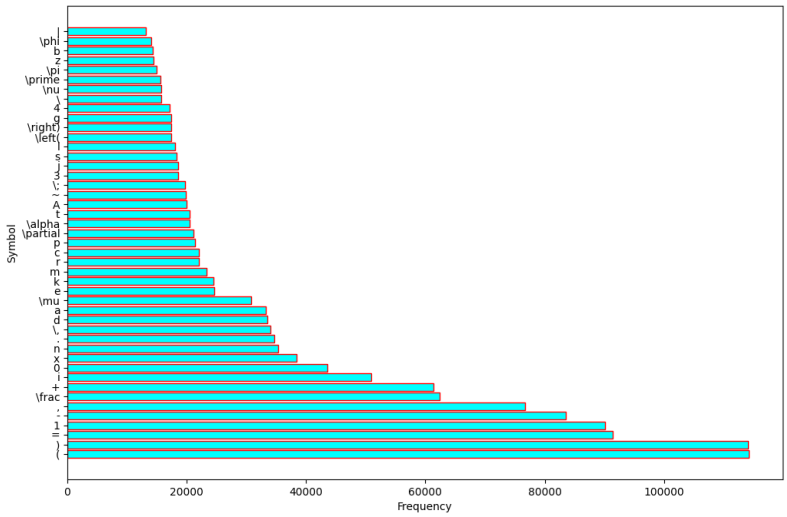


Figure 4: Dataset: Most popular symbols and frequencies.

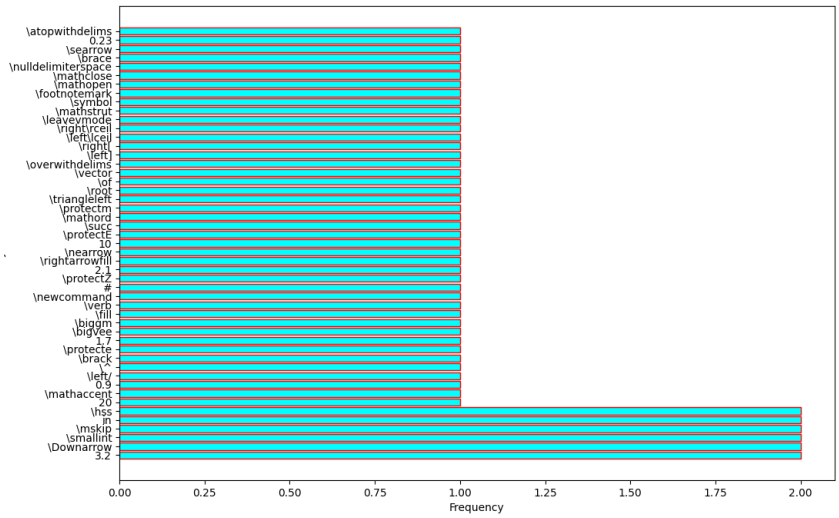


Figure 5: Dataset: Least popular symbols and frequencies.