Image2LaTeX Application: Examination of Approaches to Accuracy Assessment

Project Proposal

Gu Zhuangcheng (3035827110) - *zcgu@connect.hku.hk*
He Zixuan (3035827861) - *hzx0517@connect.hku.hk*
Deng Jiaqi (3035832490) - *jiaqideng@connect.hku.hk*

Project Objectives

Our project focuses on training a model to convert mathematical formula images into LaTeX code. We will test different approaches to improve the accuracy of the image to latex sequence task based on the state of the art models (such as DenseNet [1]). After testing and evaluating the model, we will integrate the whole project into an application with the user interface (e.g. Desktop Application) into real-world practice.

Target Application

Many people who need to write academic essays or papers will face the problem of using LaTeX to generate formulas nowadays. However, they may not have prior experience in LaTeX code programming. Furthermore, in most situations, LaTeX code is unrelated to their essay or paper topic. It is just a tool for generating a better-looks formula. Hence, it is better to have something to save time so they can focus more on their academic research topic. Hence, we want to build an application for converting formula images to LaTeX code. The expected input is a formula image containing a typed or computer-generated formula (i.e., a screenshot containing the formula the user wants to convert into LaTeX code). After putting the image, our application will generate the LaTeX code for this formula. Thus, we will train a machine-learning model to support our application. The form of application has not been finalized yet due to the uncertainty of the time cost of training and improving the model, but we will take one of the following two specific applications:

Desktop application

When users read articles using PDF viewer or through websites and want to convert the formula into LaTeX sequence, they could use our application and take screenshots of that specific formula. Our application will generate the LaTeX code

for it based on our trained model. They can also choose to use the camera on the phone to capture the image.

Mobile application

Users need to upload a formula image into the app, and our app will use the model trained to generate the LaTeX code for this formula.

Dataset

We will use the dataset IM2LATEX-100K [2], a prebuilt dataset for OpenAI's task on Kaggle. The dataset contains around 100K formulas, and images with its formulas were parsed from LaTeX sources provided by KDD Cup 2003. Those images and formulas are split into train, test, and validation sets, with information stored in three CSV files (for train, test, and validation dataset, correspondingly). Each of these three files contains two columns, with the first column storing the LaTeX formula and the second column having the name of the mathematical formula's image, which provides paths to find the actual image. The following table 1 consists of examples from the training CSV file.

Label	Image
$\widetilde{\gamma}_{\text{hopf}} \simeq \sum_{n>0} \widetilde{G}_n \frac{(-a)^n}{2^{2n-1}}$	66667cee5b.png
$(\mathcal{L}_a g)_{ij} = 0, \quad (\mathcal{L}_a H)_{ijk} = 0,$	1cbb05a562.png

Table 1: Examples from the CSV file

All formula images are stored together in another file. Each image is a fixed-size PNG image with the formula in black, and the rest of the image is transparent. The following image 1 is one example from the dataset.

$$\int_0^{r_0} r dr (\tilde{T}^{\theta}_{(0)\theta} + \tilde{T}^{r}_{(0)r}) = r_0^2 \tilde{T}^{r}_{(0)r}(r_0) = r_0^2 \left[\frac{A'^2(r_0)}{2e^2} + \frac{\alpha}{2} \Lambda'^2(r_0) \right].$$

Figure 1: Sample Image from the IM2LATEX Dataset

Methodology

Overview

Considering the similarity between our problem and the natural machine translation tasks, we will build our model based on the encoder-decoder architecture, and we will refer to the sub-models used in Wang and Liu's paper [3] on this Image2Latex problem. For our base model, we will use the convolutional neural network (CNN) with positional embeddings to encode the input of grayscale image

and using the Transformer [4] as decoder with teacher-forcing method in the decoder part. And we will try various methods to resolve the exposure bias [5] during the sequence prediction in the training process.

Encoder

For the encoder part, our task is to convert the grayscale images into a one-dimensional feature representation. The most traditional way of extracting features from 2d image is to use the regular CNN with multiple groups of convolution, pooling and batch normalizing layers. However, we also need to consider the problem of positional embedding. Regular text is always using the left-to-right location ordering, which its features can be captured by RNN, but math formula usually contains more spatial relationship and directions, for example subscript, superscript, and curly braces used in equation set. So, we also need to consider how to preserve this spatial relationship. Based on the encoder architecture from Wang and Liu [3], we mainly have two parts that we are trying to improve. First, we will test the performance of the model by using different two-dimensional positional encoding methods on the feature map of the input. The other way is to modify the structure of the CNN network. Or we can improve the architecture of the CNN network, for example, using ResNet [6] instead of VGG-Very Deep mentioned in our reference model.

Decoder

For sequential information, we would adopt the widely used RNN model as the framework. The model would take the output from the encoder and generate predictions on the LaTeX tokens based on the sequential information. Inside the RNN, we plan to use long-short term memory (LSTM) [7] for the neuron. First, it would be helpful to tackle the problem of vanishing gradient. Second, LSTM has better performance in memorizing longer sequences with past information. However, we do notice that LSTM requires a much longer time to train than traditional neuron structure. There are other variants of LSTM structure, such as bidirectional LSTM mentioned in [8]. The experiment regarding the choice of neuron structures will be conducted to compare the performance. Besides that, another key element would be the attention layer [9] which may help the model better detail with long sequences. Every hidden state in the encoder is connected to the attention layer storing the information, and these outputs are fed to the decoder with their corresponding weights. New mask attention mentioned in [10] aims to solve the over-parsing problem in the decoder. The over-parsing [11] occurs when some feature map segments are repeatedly parsed. We may also try different attention mechanisms in later experiments and share our findings based on the results from the experiments.

Project Plan

Tasks & Timeline

i. Data Exploration

a. Dataset preprocessing (*Oct. 9 - Oct. 13*)

b. Tokenize LaTeX sequence (*Oct. 12 - Oct. 15*)

c. Downsample images (*Oct. 12 - Oct. 15*)

d. Data visualization (*Oct. 14 - Oct. 17*)

ii. AI/ML Methodology Implementation

a. Construct the base model (*Oct. 13 - Oct. 20*)

b. Feasibility test (*Oct. 17 - Oct. 23*)

c. Analyse possible improvements (*Oct. 16 - Oct. 25*)

d. Model development stage 1 (*Oct. 23 - Nov. 3*)

e. Model development stage 2 (*Nov. 3 - Nov. 10*)

f. Model evaluation stage 2 (*Nov. 8 - Nov. 12*)

iii. Evaluation and Identify Limitation

a. Design the metrics between the predicted sequence and ground-truth label (*Oct. 28 - Nov. 3*)

b. Compare the built model with existing models (*Nov. 5 - Nov. 12*)

iv. Integration

a. Build application (*Nov. 12 - Nov. 20*)

b. Integral testing (*Nov. 17 - Nov. 24*)

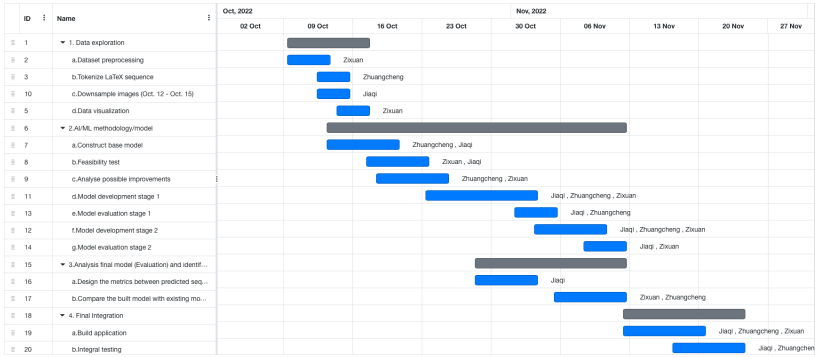

Figure 2: Timeline and Division of Labor

Platform & Tools

- ◇ **GitHub** - Synchronization and Version Control
- ◇ **HKU GPU Farm** - Cloud GPU Resources
- ◇ **Anaconda** - Package and Environment Management
- ◇ **Matplotlib** - Data visualization
- ◇ **Scikit-Learn** - Machine Learning Library
- ◇ **PyTorch** - Machine Learning Library
- ◇ **LaTeX** - Document Editor

References

[1] J. Wang, Y. Sun, and S. Wang, "Image to latex with densenet encoder and joint attention," *Procedia Computer Science*, vol. 147, pp. 374–380, 2019.

[2] Y. Deng, A. Kanervisto, J. Ling, and A. M. Rush, "Image-to-markup generation with coarse-to-fine attention," 2016.

[3] Z. Wang and J.-C. Liu, "Translating math formula images to latex sequences using deep neural networks with sequence-level training," 2019.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," p. arXiv:1706.03762, 2017.

[5] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," p. arXiv:1511.06732, 2015.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[7] S. Hochreiter and J. rgen Schmidhuber, "Long short-term memory," *Neural Computation*, pp. 1735 – 1780, 1997.

[8] A. Mnih and Y. W. Teh, "A fast and simple algorithm for training neural probabilistic language models," *Andriy Mnih and Yee Whye Teh*, 2012.

[9] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," p. arXiv:1409.0473, 2014.

[10] N. Pang, C. Yang, X. Zhu, J. Li, and X.-C. Yin, "Global context-based network with transformer for image2latex," in *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021, pp. 4650–4656.

[11] Z. Li, L. Jin, S. Lai, and Y. Zhu, "Improving attention-based handwritten mathematical expression recognition with scale augmentation and drop attention," p. arXiv:2007.10092, 2020.