Handwritten Mathematical Formula Recognition: via an End-to-End and Attention-Based Deep Learning Approach

Supervisor: Qiufeng Wang
Authors : Wei Liao, Jianghan Chen, Jiayi Liu, Lenghan Zhu, Zichuan Yang

Abstract

Recognition of handwritten formulas has been a complex task due to the wide variety of mathematical symbols and the complex combination of residing spatial properties. In our project, we firstly implement a deep learning approach via attention based encoder-decoder model that recognizes and converts input handwritten formulas from pictures into LaTex strings. We choose DenseNet as encoder and GRU as decode with coverage attention applied. We then contributed a dataset containing mathematical formula symbols of various lengths. Finally, we designed a web site to present our models.

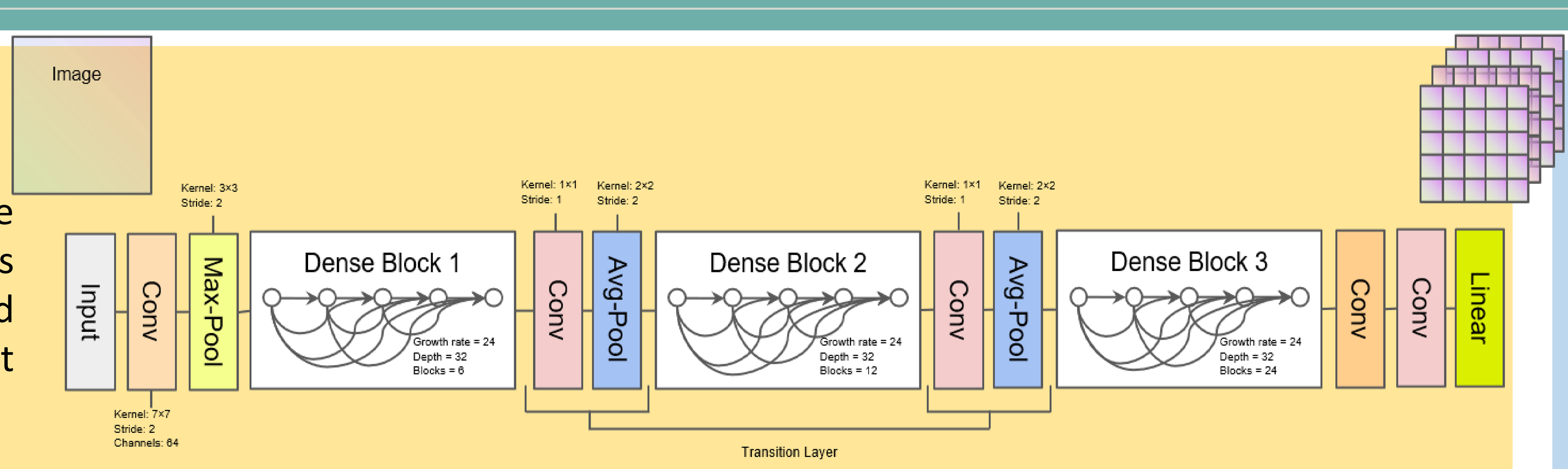
Introduction

In recent years, the booming development of deep learning has led to greatly enhanced accuracy of handwriting recognition. However, the recognition performance of existing recognition models for mathematical formulas with spatial properties has yet to be enhanced. And handwritten formula recognition is in great demand in both office software and student communities. Therefore, we implemented a website with a deep learning model that can convert handwritten formulas into latex strings. The model is based encoder-decoder with attention. The encoder is a densely connected convolutional networks (DenseNet) that maps images to high-level features. The decoder is a recurrent neural network (RNN) with gated recurrent units (GRU) that converts these features into output strings one symbol at a time. For each predicted symbol, an attention chooses the most relevant region to recognize, using the coverage model to solve over- or under-parsing problems.

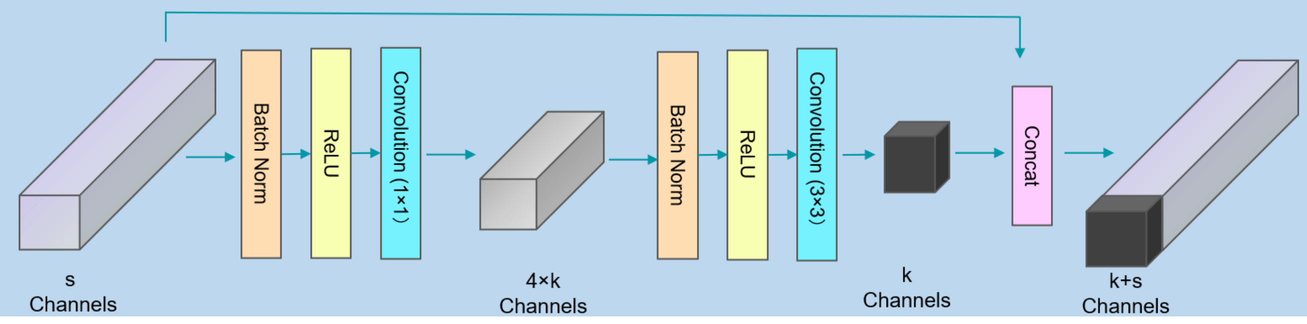
Methodology

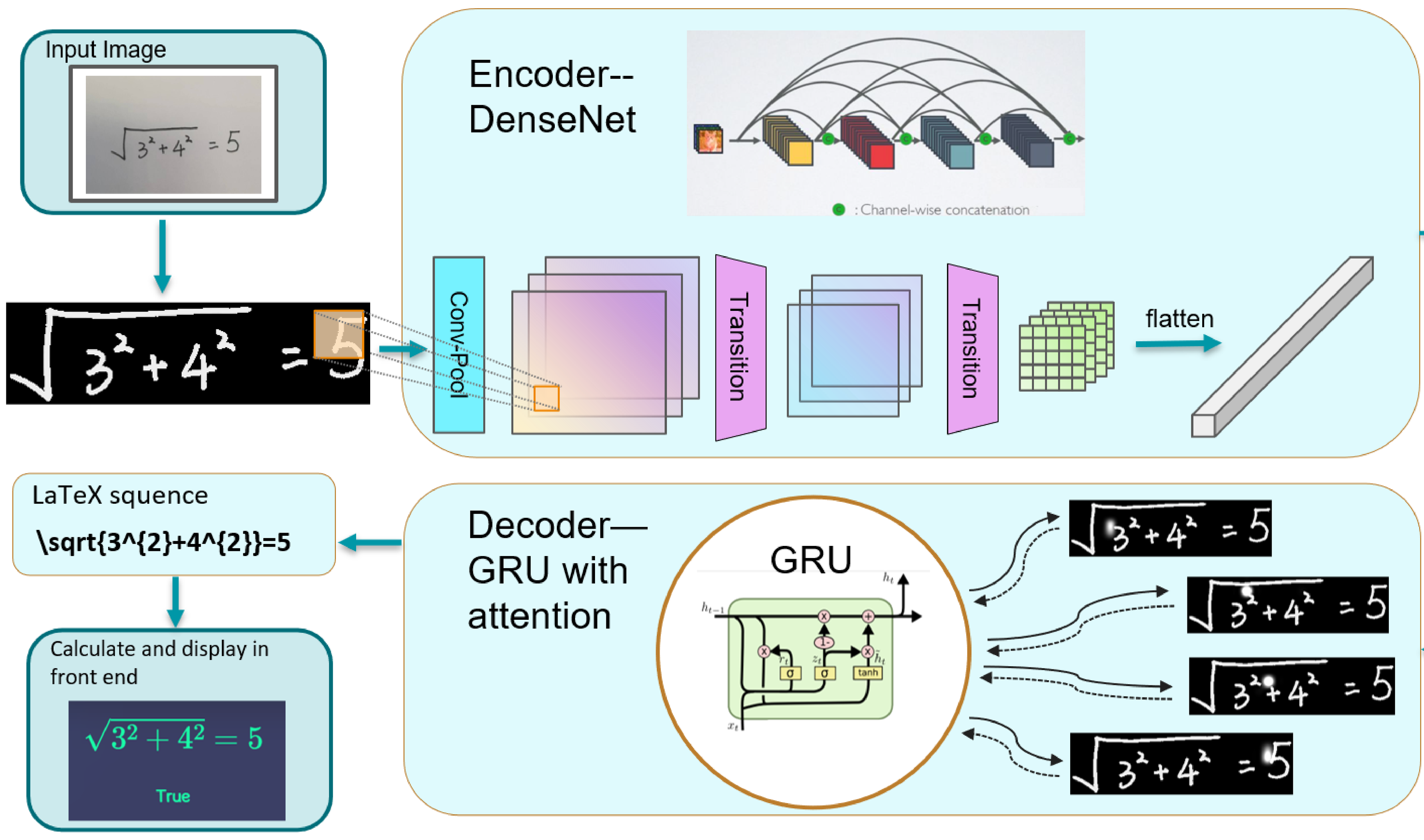
Dense Encoder

We employ three dense blocks. It strengthens feature extraction and facilitates gradient propagation.



Inside dense block we use bottleneck layers to improve computational efficiency when the input quantity increases with the number of iterations. Here the growth rate k = 24.





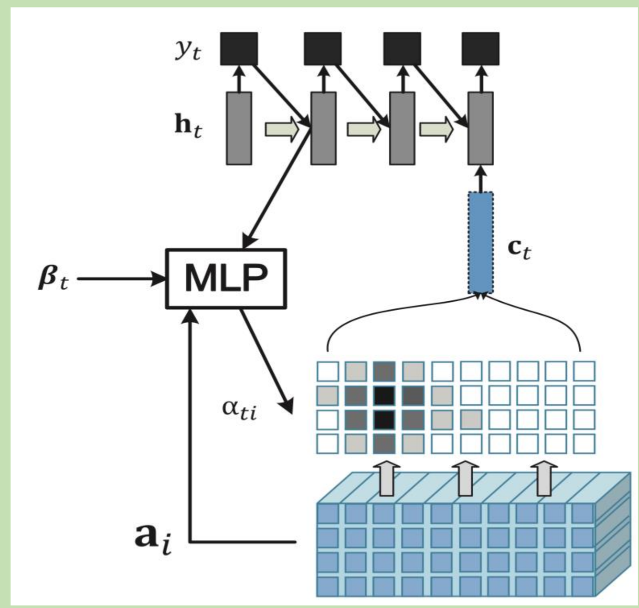
Coverage-Attention

We used the attention model based on the coverage vector as follows:

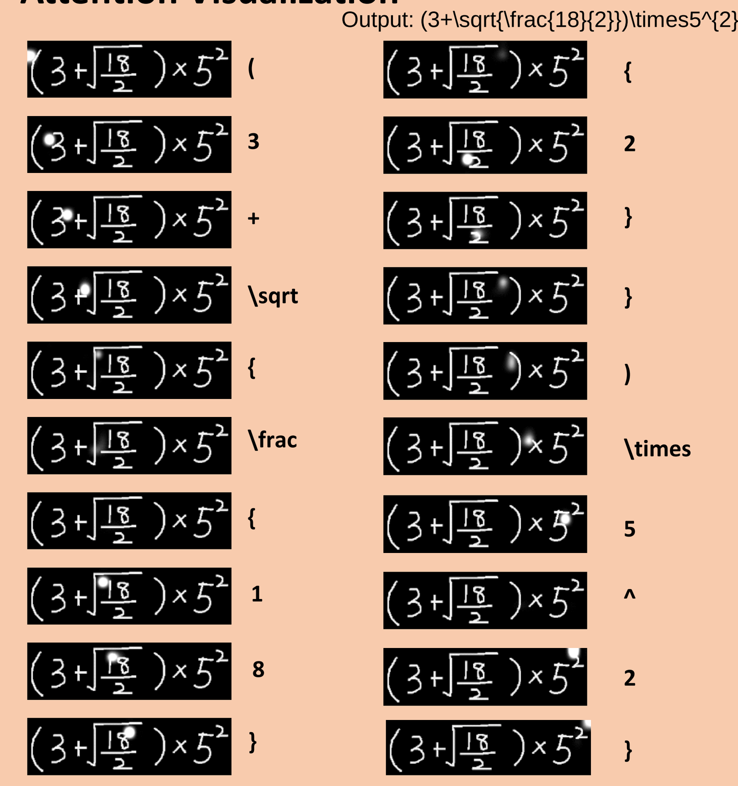
$$\boldsymbol{\beta}_t = \sum_{l}^{t-1} \boldsymbol{\alpha}_l$$

$$\boldsymbol{F} = \boldsymbol{Q} * \boldsymbol{\beta}_t$$

$$e_{ti} = \boldsymbol{\nu}_a^{\mathrm{T}} \tanh(\boldsymbol{W}_a \boldsymbol{h}_{t-1} + \boldsymbol{U}_a \boldsymbol{a}_i + \boldsymbol{U}_f \boldsymbol{f}_i)$$



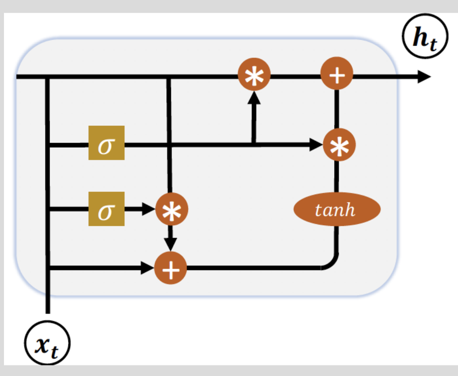
Attention Visualization



At each time step t, an MLP combines the hidden state $\boldsymbol{h}_{t-1}$ and all the annotation vectors $\boldsymbol{a}_i$ with past alignment information $\boldsymbol{\beta}_t$ to compute the attention weights α_{ti}

Decoder

We employ GRU as the decoder because it is an improved version of RNN which can alleviate the vanishing and exploding gradient problems.



A context vector $\boldsymbol{c}_t$ is to address the variable-length annotation. It is computed via variable-length annotations $\boldsymbol{a}_i$

$$\boldsymbol{c}_t = \sum_{i=1}^{L} \alpha_{ti} \boldsymbol{a}_i$$

Here, the attention weights α_{ti} will make decoder to know which part of the image is the suitable place to attend to generate the next predicted symbol and then assign a higher weight.

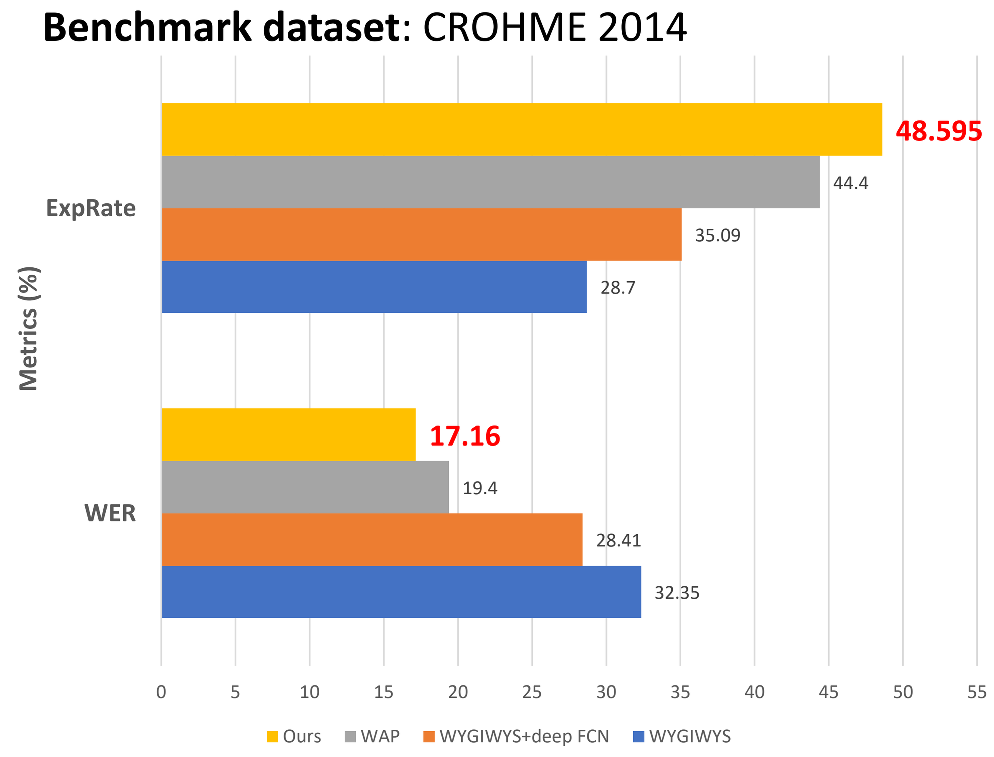
The probability of each predicted symbol is computed by:

$$p(y_t | \boldsymbol{y}_{t-1}, X) = g(\boldsymbol{W}_o h(\boldsymbol{E}\boldsymbol{y}_{t-1} + \boldsymbol{W}_s \boldsymbol{s}_t + \boldsymbol{W}_c \boldsymbol{c}_t))$$

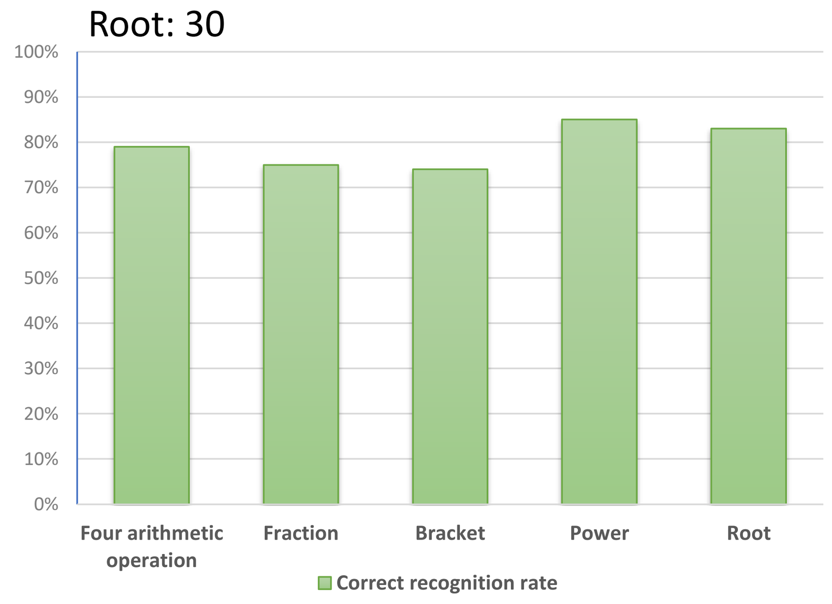
Result

Evaluation of the recognition performance (in %) comparison for different models and different types of formulas

ExpRate: expression level (Correctness/Total)
WER: word level
((substitutions+deletions+insertions)/Total)
Benchmark dataset: CROHME 2014



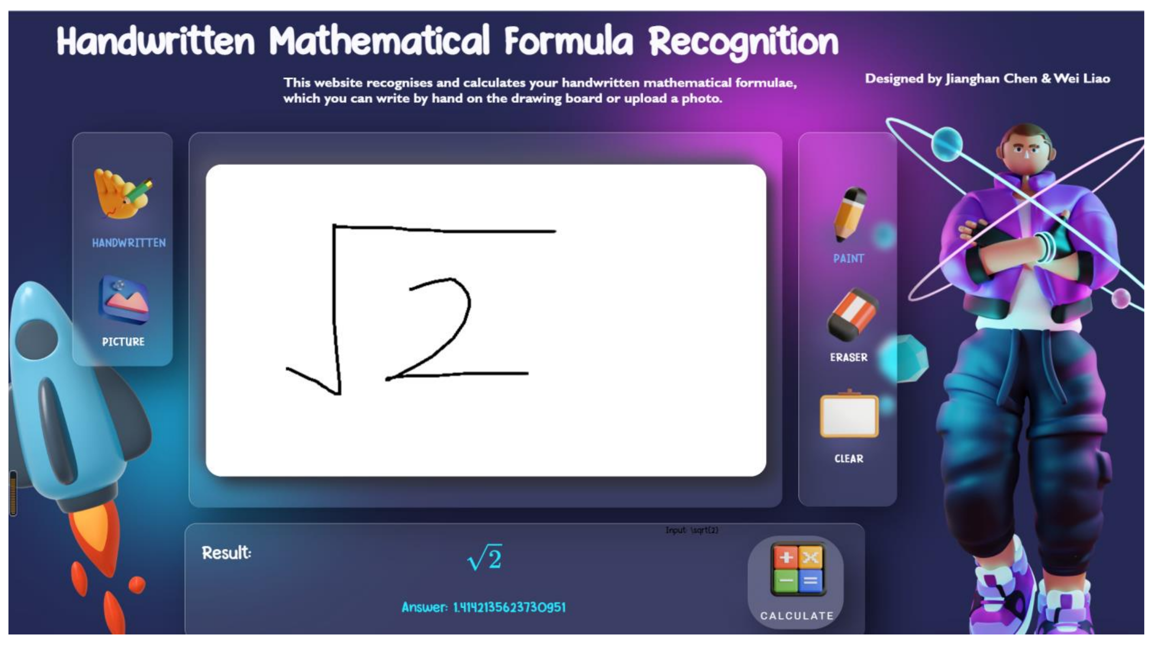
Test in 120 images containing commonly used type formulas
Number of images containing:
Four arithmetic: 60
Fraction: 30
Bracket: 60
Power: 30
Root: 30



Application

We have designed a website and deployed it to the server to represent our model.

Using technology: Django+HTML/CSS/JS+JQuery



You can experience it at the following URL or QR code (recommended for computers or tablets)

http://39.106.229.39:8000/

Conclusion

We have three main contributions. Firstly, we used a deep learning model based on DenseNet+Attention+GRU to recognize handwritten mathematical formulae and showed good performance in the Chrome2014 dataset. Secondly, we produced a formula dataset with a sample size of 1000 for training and testing the model. Finally we designed a front-end website to demonstrate the model. In future work, we will optimize it for different people's handwriting fonts and to improve the error rate.

Reference

[1] J. Zhang, J. Du and L. Dai, "Multi-Scale Attention with Dense Encoder for Handwritten Mathematical Expression Recognition," 2018 24th International Conference on Pattern Recognition (ICPR), 2018, pp. 2245-2250, doi: 10.1109/ICPR.2018.8546031.
[2] Huang G, Liu Z, Weinberger K Q, et al. Densely connected convolutional networks[J]. arXiv preprint arXiv:1608.06993, 2016.