Image To LaTeX: A Neural Network Approach

Junwen Zheng*, Zhengqing Zhou†, Zhongnan Hu‡
* Stanford CEE, † Stanford Math, ‡ Stanford ME
Email: {junwenz, zqzhou, chrisnan}@stanford.edu

Introduction

We aimed to build an Optical Character Recognition for math expressions by neural network. Specifically, we were seeking for a supervised model that can learn to produce correct LaTeX code from an image, without any knowledge of underlying language, and was simply trained end-to-end on real-world data.

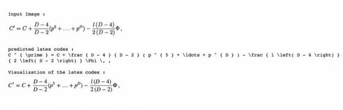

Figure 1: Example of input image and output LaTeX codes

Data Prepossessing

(1) We used the raw dataset IM2LATEX-100K in [1] that each image contained a LaTeX formula rendered on a full page.

(2) We cropped the formula area, and group images of similar sizes to facilitate batching.

(3) We divided the dataset into train ($\approx$ 80,000 images), validation ($\approx$ 8,000 images) and test set ($\approx$ 8,000 images).

Observation & Motivation

(1) The following result from the Harvard team [1] indicated the difficulty in detecting the tiny symbols in math formula.

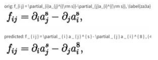

Figure 2: The prediction messed up the superscript 's' by '8'

(2) A possible reason might be the poor performance of CNN encoder [1]. Information of the tiny symbols was lost after several convolutions and max-poolings.

(3) This observation motivated us to use DenseNet [2] architecture which contains shorter connections between layers close to the input and those close to the output.

DenseNet

DenseNet model consists of two parts:

(1) Dense block
There were bottleneck layers with grow rate of 16/32/32/32 (4-Denseblock model) or 16/32/64 (3-Denseblock model) in each block after batch normalization.
Each layer took all preceding feature-maps as input and concatenated into a tensor.

(2) Transition layer
The layers between two adjacent dense blocks were referred to as transition layers.
Due to small model size, we didn't compression feature-map via convolution.
Batch normalization was applied on the output of denseblock.
Max poolings were applied with kernel (2,2) and stride (2,2),(2,2), (2,1) and (1,2) for 4 dense block case.

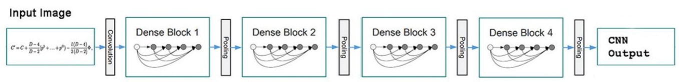

Figure 3: Deep DenseNet with four dense blocks

Model & Approach

(1) Densely Connected Network: We inputed images via DenseNet and got the output, whose shape was (batch size, 1/8 $\times$ original height, 1/8 $\times$ original width, 512).

(2) Encoder: We used RNN decoder, getting initial state came from the output of DenseNet, with gated recurrent unit (GRU) to learn learn long-term dependencies.

(3) Decoder: Among sequence-to-sequence models, we chose the attention model to focus on contents which might be useful for prediction.

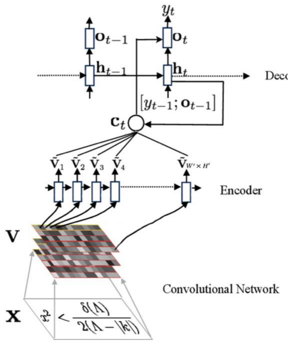

Figure 4: Network structure

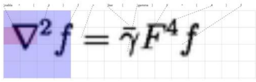

Figure 5: Attention model

learning rate	0.001/0.0001
Numbers of feature maps	512
GRU cell size	256
embedding size	80
attention size	256
batch size	20

Table 1: Hyper parameters

Results & Discussion

To evaluate the result, we tracked the accuracy of predicted latex codes and the true latex codes by the BLEU score and Edit Distance.

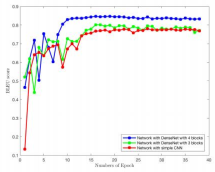

Figure 6: BLEU score

Model	BLEU score	Edit distance
[1]CNNEnc	0.75	0.61
Baseline CNN	0.77	0.76
DenseNet(3 blocks)	0.80	0.79
DenseNet(4 blocks)	0.84	0.85

Table 2: Experiment results

Discussion

(1) The results of DenseNet models were better than CNNEnc[1] and baseline CNN model, because bottleneck layers were closely connected and more features from shallower layers could be directly transferred into deeper layers.

(2) The training for DenseNet model was more efficient compared with baseline CNN model, because of parameter efficiency of densenet.

Future works

(1) We'd like to apply beam search to improve the RNN network, which as an approximate search often works far better than the greedy approach.

(2) Our research can be scaled from printed mathematical formulas images to the hand written mathematical formulas images.

References

- [1] Yuntian Deng and Anssi Kanervisto and Alexander M. Rush. *What You Get Is What You See: A Visual Markup Decompiler.* arXiv preprint arXiv:1609.04938, 2016.
- [2] Gao Huang and Zhuang Liu and Kilian Q. Weinberger. *Densely Connected Convolutional Networks.* arXiv preprint arXiv:1608.06993, 2016.