ConvMath : A Convolutional Sequence Network for Mathematical Expression Recognition

Zuoyu Yan, Xiaode Zhang, Liangcai Gao, Ke Yuan and Zhi Tang

Wangxuan Institute of Computer Technology
Peking University

December, 2020

ConvMath

ConvMath

Introduction

Main task:

- Converts the mathematical expression description in an image into a LaTeX sequence

Math expressions:

- Exhibit complicated 2-D layout(CONTAIN, ABOVE, BELOW, etc.)
- Variant scales: caused by symbol position or input
- Symbols similar in expression: α and a, π and $\prod$
- Complexity of solution for mathematical expression increases dramatically with the number of math symbols

Introduction

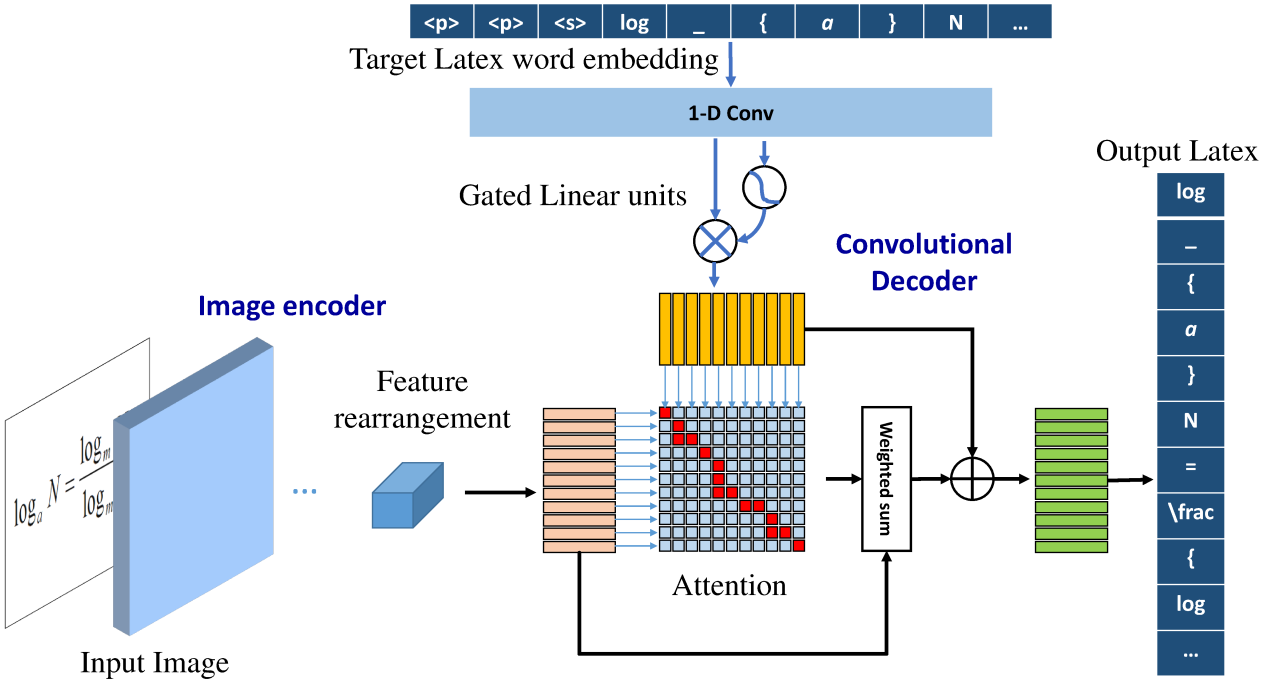
Motivation:

- Encoder-decoder model has been applied to previous works: WAP[1]
- Trained end2end: no need to define heuristic grammar rules
- Residual encoder: combine high-level and low-level features to handle scale variance
- Decoder with attention mechanism: focus on the most relevant part of math presentations
- Convolutional based: for speed up

[1] Zhang J, Du J, Dai L. Multi-scale attention with dense encoder for handwritten mathematical expression recognition[C]//2018 24th international conference on pattern recognition (ICPR). IEEE, 2018: 2245-2250.

ConvMath

Network architecture



- Input: grey scale image: $X \in R^{W*H}$
- Output: Latex sequence $Y = \{y_1, y_2, ..., y_T\}$
- Task: $\theta^* = argmax_\theta \sum_D logp(Y|X)$ where θ denotes the parameters of the model

Network architecture

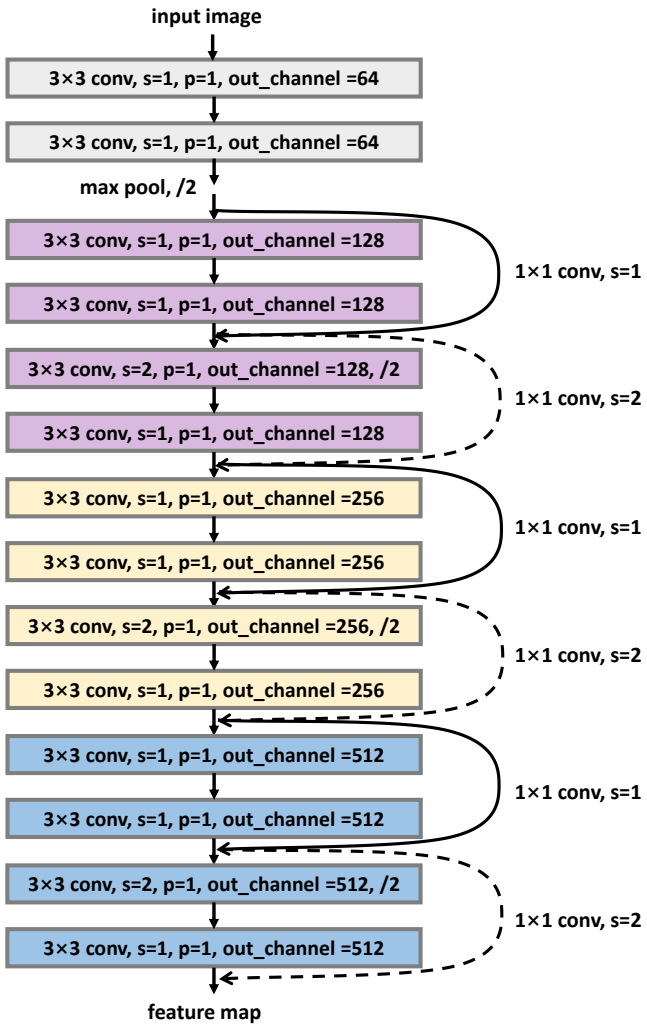
Residual encoder:

- Input: grey scale image: $X \in R^{W*H}$
- Output: feature map $\in R^{W'*H'*D}$
- Rearranged output: a sequence of feature vectors $V = \{v_1, v_2, ..., v_{W'*H'}\}$ where $v_i \in R^D$
- Rearrangement may break out spatial dependency: attention can focus on the most relevant part

Network architecture

Residual encoder:

- Can be viewed as combination of low and high level feature, which is both beneficial to modeling 2-D relationships and preserving detailed information
- Easy to optimize and keep the capacity at the same time
- Consists of six residual blocks as shown
- A 1*1 convolution to match the dimensions(solid line) and feature map sizes(dotted line)



Network architecture

Convolutional decoder:

- Input: feature vectors $V = \{v_1, v_2, ..., v_{W'*H'}\}$
- Output: Latex sequence $Y = \{y_1, y_2, ..., y_T\}$
- Entirely convolutional: both the size of image and Latex string are not fixed

Network architecture

Latex embedding and position embedding:

- Latex embedding: $W = \{w_1, w_2, ..., w_n\}$, where $w_i \in R^D$, same setting as [2]
- Position embedding(important for convolutional decoder): $P = \{p_1, p_2, ..., p_N\}$, where $p_i \in R^D$, same setting as [3]
- Final representation: $G = \{g_1, g_2, ..., g_N\} = \{w_1 + p_1, w_2 + p_2, ..., w_N + p_N\}$

[2] Sennrich R, Haddow B. Linguistic input features improve neural machine translation[J]. arXiv preprint arXiv:1606.02892, 2016.

[3] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in neural information processing systems. 2017: 5998-6008.

Network architecture

Convolutional decoder:

- Stack multiple(L) basic blocks
- Output of the l-th block: $h_l = \{h_1^l, h_2^l, ..., h_N^l\}$
- Residual connection between blocks: $h^l = conv(h^{l-1}) + h^{l-1}$
- Final output probability:
  $p(y_{i+1}|y_1, ..., y_i, V) = softmax(Wh_i^L + b) \in R^K$ where W and b are weight and bias of the linear mapping layer, K is the size of vocabulary
- Minimize: $L = -\frac{1}{|D|}\sum_D \sum_{i=1}^N logp(y_i|y_{<i}, X)$

Network architecture

Basic decoder block:

- Consists of a 1-dimensional convolution and a subsequent gated linear unit(GLU)
- 1-dimensional convolution: capture the dependencies among Latex symbols, with weight $W \in R^{2D*kD}$ and bias $b \in R^{2D}$
- Input: k continuous elements in a Latex string, output: 2D-dimensional vector $M \in R^{2D} = [A; B]$ where $A \in R^D, B \in R^D$
- GLU: to select the important part: $GLU(M) = A \otimes \sigma(B)$ where $\otimes$ is the point-wise multiplication and σ is sigmoid function

Network architecture

Attention mechanism:

- To focus on the most relevant part
- Content vector: $c_i^l = \sum_{j=1}^{W'*H'} a_{ij}^l v_j$, here c_i^l is the content vector of the l-th decoder layer corresponding to the i-th state
- Attention score: $a_{ij}^l = \frac{exp(d_i^l, v_j)}{\sum_{t=1}^{W'*H'} exp(d_i^l, v_t)}$
- Decoder state summary: $d_i^l = W_d^l h_i^l + b_d^l + g_i$, which combines the current layer output and representation of the previous target element g_i
- $c_i^l + h_i^l$ as the input of the next layer

Network architecture

Attention mechanism:

- Applied to each decoder layer
- Alleviate the problem of lacking coverage
- Coverage: the overall alignment information that indicates whether a local region of the feature vector has been translated
- Under/Over parsing: some feature vectors are not parsed/ generated multiple times
- Previous attention is accumulated: achieve the tracking of past alignment information

ConvMath

Experiments

TABLE I: Experimental results on IM2LATEX-100K.

Method	BLEU	time($s/batch$)	Edit Distance	Exact Match
WYGIWYS [18]	87.73	0.129	87.60	79.88
WAP [1]	88.21	0.135	89.58	82.08
ConvMath	88.33	0.083	90.80	83.41

- Dataset: IM2LATEX-100K, contains Latex expressions from over 60000 papers from arxiv
- Training/validation/test set: 65995/8181/8301 expressions
- Symbol dictionary: 583, embedding size: 512
- Evaluation: BLEU score, column-wise edit distance, exact match accuracy, the elapsed time to finish a forward inference for a batch(batch size 10)

Experiments

TABLE II: Contributions of different parts in the proposed network.

Method	BLEU
WYGIWYS [18]	87.73
WAP [1]	88.21
ConvMath_SimpleEncoder	80.72
ConvMath(3 decoder layers)	84.81
ConvMath(5 decoder layers)	87.61
ConvMath(7 decoder layers)	88.33
ConvMath(9 decoder layers)	88.04

- Residual encoder: ConvMath_simple Encoder and ConvMath(7 decoder layers): combine high-level and low-level features
- The performance with regard to the depth of decoder: increases first(large receptive field) and drops after(risk of overfitting)

Case study

Image	$M_g = M_{c_1} M_{c_2} M_{c_3} M_{c_4} M_{c_5} M_{r=\infty} = 1$
Ground truth	M _ { g } = M _ { c _ { 1 } } M _ { c _ { 2 } } M _ { c _ { 3 } } M _ { c _ { 4 } } M _ { c _ { 5 } } M _ { r = \infty } = 1
WYGIWYS	M _ { g } = M _ { c _ { 1 } } M _ { c _ { 2 } } M _ { c _ { 3 } } M _ { c _ { 2 } } M _ { c _ { 5 } } M _ { r = \infty } = 1 \qquad \qquad
ConvMath	M _ { g } = M _ { c _ { 1 } } M _ { c _ { 2 } } M _ { c _ { 3 } } M _ { c _ { 4 } } M _ { c _ { 5 } } M _ { r = \infty } = 1
Image	$V(z, \bar{z}) = e^{-q\Phi(z)} e^{i\alpha\cdot H} e^{i(P_R\cdot X_R - P_L\cdot X_L)}$,
Ground truth	V (z , \bar { z }) = e ^ { - q \Phi (z) } e ^ { i \alpha \cdot H } e ^ { i (P _ { R } \cdot X _ { R } - P _ { L } \cdot X _ { L }) } \; ,
WYGIWYS	V (z , \bar { z }) = e ^ { - q \Phi (z) } e ^ { i \alpha \cdot H } e ^ { i (P _ { R } \rightarrow X _ { R } - P _ { L } X _ { L }) } \; , \hspace { 1 c m }
ConvMath	V (z , \bar { z }) = e ^ { - q \Phi (z) } e ^ { i \alpha \cdot H } e ^ { i (P _ { R } \cdot X _ { R } - P _ { L } \cdot X _ { L }) } \;
Image	$R(e_1) = \epsilon^{-J_{67}+J_{89}}, \quad R(e_2) = \epsilon^{J_{45}-J_{89}}.$
Ground truth	R (e _ { 1 }) = \epsilon ^ { - J _ { 6 7 } + J _ { 8 9 } } , \quad R (e _ { 2 }) = \epsilon ^ { J _ { 4 5 } - J _ { 8 9 } } .
WYGIWYS	R (e _ { 1 }) = \epsilon ^ { - J _ { 0 } + J _ { 8 } } , \quad R (e _ { 2 }) = \epsilon ^ { J _ { 3 } - J _ { 8 } } .
ConvMath	R (e _ { 1 }) = \epsilon ^ { - J _ { 0 } + J _ { 8 9 } } , \quad R (e _ { 2 }) = \epsilon ^ { I _ { 4 } } - J _ { 8 9 } } .

- Errors: highlighted in red
- Over parsing rarely happens
- Under parsing is common(the third example)
- Future direction: strengthen the ability to deal with under parsing problems

ConvMath

Conclusion

Contribution:

- Propose a convolution based model which achieves SOTA results and much higher speed
- Residual encoder to combine high-level and low-level features
- Combine multi-layer attention mechanism with the decoder, which solves the problem of lacking coverage

Future directions:

- Evaluate the network on other datasets like handwritten mathematical expression datasets.
- Apply the network to other tasks such as image caption generation, musical score recognition et al.

Thanks for listening!