Multi-Scale Attention with Dense Encoder for Handwritten Mathematical Expression Recognition

Jianshu Zhang, Jun Du and Lirong Dai

National Engineering Laboratory for Speech and Language Information Processing

University of Science and Technology of China, Hefei, Anhui, P. R. China

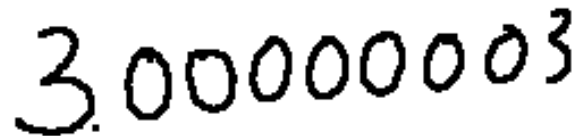Email: xysszjs@mail.ustc.edu.cn, jundu@ustc.edu.cn, lrdai@ustc.edu.cn

Abstract—Handwritten mathematical expression recognition is a challenging problem due to the complicated two-dimensional structures, ambiguous handwriting input and variant scales of handwritten math symbols. To settle this problem, recently we propose the attention based encoder-decoder model that recognizes mathematical expression images from two-dimensional layouts to one-dimensional LaTeX strings. In this study, we improve the encoder by employing densely connected convolutional networks as they can strengthen feature extraction and facilitate gradient propagation especially on a small training set. We also present a novel multi-scale attention model which is employed to deal with the recognition of math symbols in different scales and restore the fine-grained details dropped by pooling operations. Validated on the CROHME competition task, the proposed method significantly outperforms the state-of-the-art methods with an expression recognition accuracy of 52.8% on CROHME 2014 and 50.1% on CROHME 2016, by only using the official training dataset.

I. INTRODUCTION

Mathematical expressions are indispensable for describing problems in maths, physics and many other fields. Meanwhile, people have begun to use handwritten mathematical expressions as one natural input mode. However, machine recognition of these handwritten mathematical expressions is difficult and exhibits three distinct challenges [1], i.e., the complicated two-dimensional structures, enormous ambiguities coming from handwriting input and variant scales of handwritten math symbols.

Handwritten mathematical expression recognition comprises two major problems [2]: symbol recognition and structural analysis. The two problems can be solved sequentially [3] or globally [4]. However, both conventional sequential and global approaches have the following limitations: 1) explicit or implicit symbol segmentation brings many difficulties; 2) the structural analysis is commonly based on two-dimensional context free grammar [5], which requires priori knowledge to define a math grammar; 3) the complexity of parsing algorithms increases with the size of math grammar.

In recent research of deep learning, a novel attention based encoder-decoder model has been proposed [6], [7]. Its general application in machine translation [8], speech recognition [9], character recognition [10], [11] and image captioning [12] inspires researchers that mathematical expression recognition can also be one proper application [13]–[16]. More specifically, [13] proposed a model namely WAP. The WAP learns to encode input expression images and decode them into LaTeX

Fig. 1. An incorrectly recognized example of handwritten mathematical expression due to the under-parsing problem: the decimal point "." is missed in the predicted LaTeX notation.

strings. The encoder is a convolutional neural network (CNN) [17] based on VGG architecture [18] that maps images to high-level features. The decoder is a recurrent neural network (RNN) [19] with gated recurrent units (GRU) [20] that converts these high-level features into output strings one symbol at a time. For each predicted symbol, an attention model built in the decoder scans the entire input expression image and chooses the most relevant region to describe a math symbol or an implicit spatial operator. Compared with conventional approaches for handwritten mathematical expression recognition, the attention based encoder-decoder model possesses three distinctive properties: 1) It is end-to-end trainable; 2) It is data-driven, in contrast to traditional systems that require a predefined math grammar; 3) Symbol segmentation can be automatically performed through attention model.

In this study, we still focus on offline handwritten mathematical expression recognition and report our recent progress on WAP model [13]. The main contribution is in two aspects. Firstly, we improve the CNN encoder by employing a novel architecture called densely connected convolutional networks (DenseNet) [21]. The DenseNet has shown excellent performance on image classification task as it strengthens feature extraction and facilitates gradient propagation. Secondly, we present a novel multi-scale attention model to deal with the problems caused by pooling operations. Although pooling layers are essential parts of convolutional networks, they shrink the size of feature maps, yielding decrease of resolution. Because the scales of handwritten math symbols vary severely, the fine-grained details of extracted feature maps are especially important in handwritten mathematical expression recognition, which are lost in low-resolution feature

maps. For example, in Fig. 1, the decimal point is very close to math symbol "3" and its scale is much smaller than its adjacent symbols. After many pooling layers, the visual information of the decimal point is gone, which leads to an under-parsing problem. To implement the multi-scale attention model, we propose a multi-scale dense encoder that will provide both low-resolution features and high-resolution features. The low-resolution features capture a larger receptive field and are more semantic while the high-resolution features restore more fine-grained visual information. The decoder then attends to both low-resolution and high-resolution features for predicting output LaTeX strings.

II. METHODOLOGY

In this section, we first make a brief summarization of DenseNet since our encoder is based on densely connected convolutional blocks. Then we introduce the classic attention based encoder-decoder framework. Finally, we extend DenseNet by introducing a multi-scale dense encoder and describe the implementation of multi-scale attention model in detail.

A. Dense Encoder

The main idea of DenseNet is to use the concatenation of output feature maps of preceding layers as the input of succeeding layers. As DenseNet is composed of many convolution layers, let $H_l(\cdot)$ denote the convolution function of the lth layer, then the output of layer l is represented as:

$$\mathbf{x}_l = H_l([\mathbf{x}_0; \mathbf{x}_1; \ldots; \mathbf{x}_{l-1}]) \tag{1}$$

where $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_l$ denote the output features produced in layers $0, 1, \ldots, l$, ";" denotes the concatenation operation of feature maps. This iterative connection enables the network to learn shorter interactions cross different layers and reuses features computed in preceding layers. By doing so the DenseNet strengthens feature extraction and facilitates gradient propagation.

An essential part of convolutional networks is pooling layers, which is capable of increasing receptive field and improving invariance. However, the pooling layers disenable the concatenation operation as the size of feature maps changes. Also, DenseNet is inherently memory demanding because the number of inter-layer connections grows quadratically with depth. Consequently, the DenseNet is divided into multiple densely connected blocks as shown in Fig. 2. A compression layer is appended before each pooling layer to further improve model compactness.

B. Decoder

We employ GRU as the decoder because it is an improved version of simple RNN which can alleviate the vanishing and exploding gradient problems [22], [23]. Given input $\mathbf{x}_t$, the GRU output $\mathbf{h}_t$ is computed by:

$$\mathbf{h}_t = \text{GRU}\left(\mathbf{x}_t, \mathbf{h}_{t-1}\right) \tag{2}$$

and the GRU function can be expanded as follows:

$$\mathbf{z}_t = \sigma(\mathbf{W}_{xz}\mathbf{x}_t + \mathbf{U}_{hz}\mathbf{h}_{t-1}) \tag{3}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_{xr}\mathbf{x}_t + \mathbf{U}_{hr}\mathbf{h}_{t-1}) \tag{4}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{U}_{rh}(\mathbf{r}_t \otimes \mathbf{h}_{t-1})) \tag{5}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \otimes \mathbf{h}_{t-1} + \mathbf{z}_t \otimes \tilde{\mathbf{h}}_t \tag{6}$$

where σ is the sigmoid function and $\otimes$ is an element-wise multiplication operator. $\mathbf{z}_t$, $\mathbf{r}_t$ and $\tilde{\mathbf{h}}_t$ are the update gate, reset gate and candidate activation, respectively.

Assuming the output of CNN encoder is a three-dimensional array of size $H \times W \times C$, consider the output as a variable-length grid of L elements, $L = H \times W$. Each of these elements is a C-dimensional annotation that corresponds to a local region of the image.

$$\mathbf{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_L\} \ , \ \mathbf{a}_i \in \mathbb{R}^C \tag{7}$$

Meanwhile, the GRU decoder is employed to generate a corresponding LaTeX string of the input mathematical expression. The output string $\mathbf{Y}$ is represented by a sequence of one-hot encoded symbols.

$$\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_T\} \ , \ \mathbf{y}_i \in \mathbb{R}^K \tag{8}$$

where K is the number of total symbols in the vocabulary and T is the length of LaTeX string.

Note that, both the annotation sequence $\mathbf{A}$ and the LaTeX string $\mathbf{Y}$ are not fixed-length. To address the learning problem of variable-length annotation sequences and associate them with variable-length output sequences, we attempt to compute an intermediate fixed-length vector $\mathbf{c}_t$, namely context vector, at each decoding step t. The context vector $\mathbf{c}_t$ is computed via weighted summing the variable-length annotations $\mathbf{a}_i$:

$$\mathbf{c}_t = \sum\nolimits_{i=1}^{L} \alpha_{ti}\mathbf{a}_i \tag{9}$$

Here, the weighting coefficients α_{ti} are called attention probabilities and they will make decoder to know which part of input image is the suitable place to attend to generate the next predicted symbol and then assign a higher weight to the corresponding local annotation vectors $\mathbf{a}_i$. After computing the intermediate fixed-length context vector, we then generate the LaTeX string one symbol at a time. By doing so, the problem of associating variable-length annotation sequences with variable-length output LaTeX strings is addressed.

The probability of each predicted symbol is computed by the context vector $\mathbf{c}_t$, current decoder state $\mathbf{s}_t$ and previous target symbol $\mathbf{y}_{t-1}$ using the following equation:

$$p(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{X}) = g\left(\mathbf{W}_o h(\mathbf{E}\mathbf{y}_{t-1} + \mathbf{W}_s\mathbf{s}_t + \mathbf{W}_c\mathbf{c}_t)\right) \tag{10}$$

where $\mathbf{X}$ denotes input mathematical expression images, g denotes a softmax activation function, h denotes a maxout activation function, let m and n denote the dimensions of embedding and GRU decoder state respectively, then $\mathbf{W}_o \in \mathbb{R}^{K \times \frac{m}{2}}$ and $\mathbf{W}_s \in \mathbb{R}^{m \times n}$, $\mathbf{E}$ denotes the embedding matrix.
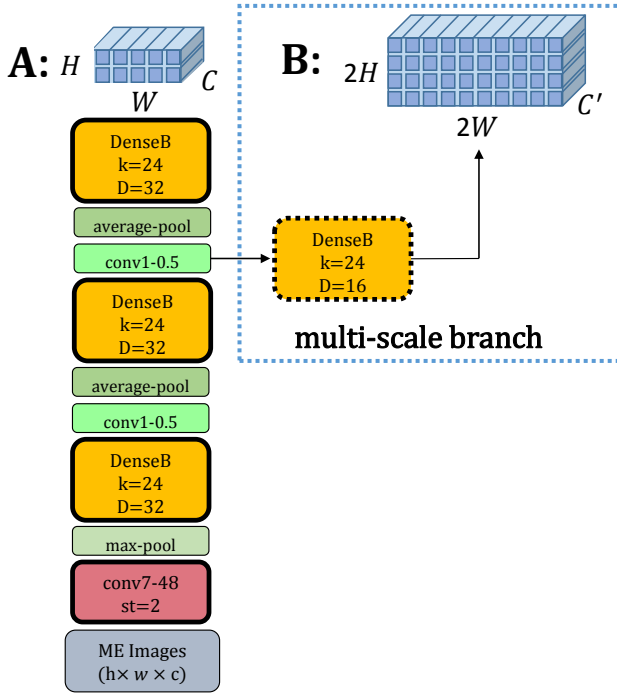
Fig. 2. Architecture of multi-scale dense encoder. The left part is the main branch while the right part is the multi-scale branch.

C. Multi-Scale Attention with Dense Encoder

1) Multi-Scale Dense Encoder: To implement the multi-scale attention model, we first extend the single-scale dense encoder into multi-scale dense encoder. As illustrated in Fig. 2, our dense encoder consists of two branches, i.e., except the main branch which produces low-resolution annotations $\mathbf{A}$, our dense encoder has another multi-scale branch that produces high-resolution annotations $\mathbf{B}$. The multi-scale branch is extended before the last pooling layer of the main branch so that the output feature maps of multi-scale branch has a higher resolution. The high-resolution annotation is a three-dimensional array of size $2H \times 2W \times C'$, which can be represented as a variable-length grid of $4L$ elements:

$$\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_{4L}\} \ , \ \mathbf{b}_i \in \mathbb{R}^{C'} \qquad (11)$$

where L is the length of annotation sequence A. Intuitively, we can extend several multi-scale branches before every pooling layer but such operation brings too much computational cost as the size of feature maps becomes too large.

As for the implementation details of dense encoder, we employ three dense blocks in the main branch as described by yellow rectangles in Fig. 2. Before entering the first dense block, a 7×7 convolution (stride is 2×2) with 48 output channels is performed on the input expression images, followed by a 2×2 max pooling layer. Each dense block is titled as "DenseB" because we use bottleneck layers to improve computational efficiency, i.e. a 1×1 convolution is introduced before each 3×3 convolution to reduce the input to $4k$ feature maps. The growth rate $k = 24$ and the depth

(number of convolution layers) of each block $D = 32$ which means each block has 16 1×1 convolution layers and 16 3×3 convolution layers. A batch normalization layer [24] and a ReLU activation layer [25] are performed after each convolution layer consecutively. We use 1×1 convolution followed by 2×2 average pooling as transition layers between two contiguous dense blocks. The transition layer reduces the number of feature maps of each block by half. While in the multi-scale branch, we append another dense block with bottleneck layer, $k = 24$ and $D = 16$. We investigate the depth of block in multi-scale branch ($D = 0, 8, 16, 24$) in Section IV.

2) Multi-Scale Attention Model: In this study, our decoder adopts two unidirectional GRU layers to calculate the decoder state $\mathbf{s}_t$ and the multi-scale context vector $\mathbf{c}_t$ that are both used as input to calculate the probability of predicted symbol in Eq. (10). We employ two different single-scale coverage based attention model to generate the low-resolution context vector and high-resolution context vector by attending to low-resolution annotations and high-resolution annotations respectively. As the low-resolution context vector and high-resolution context vector have the same length 1, we concatenate them to produce the multi-scale context vector:

$$\hat{\mathbf{s}}_t = \mathrm{GRU}\left(\mathbf{y}_{t-1}, \mathbf{s}_{t-1}\right) \qquad (12)$$

$$\mathbf{cA}_t = f_{\mathrm{catt}}\left(\mathbf{A}, \hat{\mathbf{s}}_t\right) \qquad (13)$$

$$\mathbf{cB}_t = f_{\mathrm{catt}}\left(\mathbf{B}, \hat{\mathbf{s}}_t\right) \qquad (14)$$

$$\mathbf{c}_t = [\mathbf{cA}_t; \mathbf{cB}_t] \qquad (15)$$

$$\mathbf{s}_t = \mathrm{GRU}\left(\mathbf{c}_t, \hat{\mathbf{s}}_t\right) \qquad (16)$$

where $\mathbf{s}_{t-1}$ denotes the previous decoder state, $\hat{\mathbf{s}}_t$ is the prediction of current decoder state, $\mathbf{cA}_t$ is the low-resolution context vector at decoding step t, similarly $\mathbf{cB}_t$ is the high-resolution context vector. The multi-scale context vector $\mathbf{c}_t$ is the concatenation of $\mathbf{cA}_t$ and $\mathbf{cB}_t$ and it performs as the input during the computation of current decoder state $\mathbf{s}_t$.

f_{catt} denotes a single-scale coverage based attention model. Take the computation of low-resolution context vector $\mathbf{cA}_t$ as an example, we parameterize f_{catt} as a multi-layer perceptron:

$$\mathbf{F} = \mathbf{Q} * \sum_{l=1}^{t-1} \boldsymbol{\alpha}_l \qquad (17)$$

$$e_{ti} = \boldsymbol{\nu}_{\mathrm{att}}^{\mathrm{T}} \tanh(\mathbf{U}_s \hat{\mathbf{s}}_t + \mathbf{U}_a \mathbf{a}_i + \mathbf{U}_f \mathbf{f}_i) \qquad (18)$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{L} \exp(e_{tk})} \qquad (19)$$

$$\mathbf{cA}_t = \sum_{i=1}^{L} \alpha_{ti} \mathbf{a}_i \qquad (20)$$

where $\mathbf{a}_i$ denotes the element of low-resolution annotation sequence $\mathbf{A}$, e_{ti} denotes the energy of $\mathbf{a}_i$ at time step t conditioned on the prediction of current decoder state $\hat{\mathbf{s}}_t$ and coverage vector $\mathbf{f}_i$. The coverage vector is initialized as a zero vector and we compute it based on the summation of all past attention probabilities. Hence the coverage vector contains the information of alignment history. We append the coverage vector in the attention model so that the decoder is capable to know which part of input image has been attended or not [13], [26]. Let n' denote the attention dimension and q denote the

number of output channels of convolution function $\mathbf{Q}$; then $\boldsymbol{\nu}_{\text{att}} \in \mathbb{R}^{n'}$, $\mathbf{U}_s \in \mathbb{R}^{n' \times n}$, $\mathbf{U}_a \in \mathbb{R}^{n' \times C}$ and $\mathbf{U}_f \in \mathbb{R}^{n' \times q}$. The high-resolution context vector $\mathbf{cB}_t$ is computed based on another coverage based attention model f_{catt} with different initialized parameters except the $\mathbf{U}_s$ transition matrix.

III. TRAINING AND TESTING DETAILS

We validated the proposed model on CROHME 2014 [27] test set and CROHME 2016 [28] test set. The CROHME competition dataset is currently the most widely used public dataset for handwritten mathematical expression recognition. The training set has 8,836 expressions including 101 math symbol classes. The CROHME 2014 test set has 986 expressions while the CROHME 2016 test set has 1,147 expressions.

A. Training

The training objective of the proposed model is to maximize the predicted symbol probability as shown in Eq. (10) and we use cross-entropy (CE) as the objective function:

$$O = -\sum_{t=1}^{T} \log p(w_t | \mathbf{y}_{t-1}, \mathbf{x}) \tag{21}$$

where w_t represents the ground truth word at time step t.

The implementation details of Dense encoder has been introduced in Section II-C1. The decoder is a single layer with 256 forward GRU units. The embedding dimension m and decoder state dimension n are set to 256. The multi-scale attention dimension n' is set to 512. The convolution kernel size for computing low-resolution coverage vector is set to 11×11 but 7×7 for high-resolution coverage vector, while their number of convolution filters are both set to 256. We utilized the adadelta algorithm [29] with gradient clipping for optimization. The best model is determined in terms of word error rate (WER) of validation set. We used a weight decay of 10^{-4} and we added a dropout layer [30] after each convolutional layer and set the drop rate to 0.2.

B. Decoding

In the decoding stage, we aim to generate a most likely LaTeX string given the input image. However, different from the training procedure, we do not have the ground truth of previous predicted symbol. Consequently, a simple left-to-right beam search algorithm [31] is employed to implement the decoding procedure. Here, we maintained a set of 10 partial hypotheses at each time step, ending with the end-of-sentence token $< eos >$. We also adopted the ensemble method [32] for improving the performance. We first trained 5 models on the same training set but with different initialized parameters. Then we averaged their prediction probabilities during the beam search process.

IV. EXPERIMENTS

We designed a set of experiments to validate the effectiveness of the proposed method for handwritten mathematical expression recognition by answering the following questions:

Q1 Is the dense encoder effective?
Q2 Is the multi-scale attention model effective?

Q3 Does the proposed approach outperform state-of-the-arts?

A. Metric

The participating systems in all of the CROHME competitions were ranked by expression recognition rates (ExpRate), i.e., the percentage of predicted mathematical expressions matching the ground truth, which is simple to understand and provides a useful global performance metric. The CROHME competition compared the competing systems not only by ExpRate but also those with at most one to three symbol-level errors. In our experiments, we first transferred the generated LaTeX strings into MathML representation and then computed these metrics by using the official tool provided by the organizer of CROHME. However, it seems inappropriate to evaluate an expression recognition system only at expression level. So we also evaluated our system at symbol-level by using WER metric.

B. Evaluation of dense encoder (Q1)

We start the proposed multi-scale attention model with dense encoder from WAP [13]. As shown in Table I, WAP achieves an ExpRate of 44.4% on CROHME 2014 test set and an ExpRate of 42.0% on CROHME 2016 test set. The WAP employs an encoder based on VGG architecture and its decoder is a unidirectional GRU equipped with coverage based attention model. Here, we only replace the VGG encoder by dense encoder with the other settings keeping unchanged and the new system is named as "Dense" in Table I. The implementation details of the dense encoder is illustrated by the main branch in Fig. 2. We can observe that the ExpRate increases about 5.7% on CROHME 2014 and 5.5% on CROHME 2016 by employing dense encoder.

TABLE I
COMPARISON OF RECOGNITION PERFORMANCE (IN %) ON CROHME 2014 AND CROHME 2016 WHEN EMPLOYING DENSE ENCODER AND MULTI-SCALE ATTENTION MODEL.

System	CROHME 2014		CROHME 2016	
	WER	ExpRate	WER	ExpRate
WAP	19.4	44.4	19.7	42.0
Dense	13.9	50.1	15.4	47.5
Dense+MSA	12.9	52.8	13.7	50.1

C. Evaluation of multi-scale attention model (Q2)

In Table I, the system "Dense+MSA" is the proposed multi-scale attention model with dense encoder. "+MSA" means that we only replace the single-scale coverage based attention model in system "Dense" by multi-scale coverage based attention model. The performance of multi-scale attention model is clear to be observed by the comparison between system "Dense" and system "Dense+MSA". The ExpRate increases from 50.1% to 52.8% on CROHME 2014 and from 47.5% to 50.1% on CROHME 2016 after the implementation of multi-scale attention model.

More specifically, in the system "Dense+MSA", the multi-scale branch of dense encoder contains a dense block with depth $D = 16$. We choose $D = 16$ as we investigate the depth of block in multi-scale branch ($D = 0, 8, 16, 24$) by experiments. In Table II, $D = 0$ means that we simply choose the output of the last transition convolutional layer in the main branch of dense encoder as the high-resolution annotations. The performance is only slightly improved compared with system "Dense" in Table I which implies that more convolution operations are necessary to obtain more semantic high-resolution annotations. We can observe that $D = 16$ is the best setting for both test sets of CROHME 2014 and 2016. The unpleasant results of $D = 24$ indicate that too many convolution operations in the multi-scale branch can also lead to performance degradation.
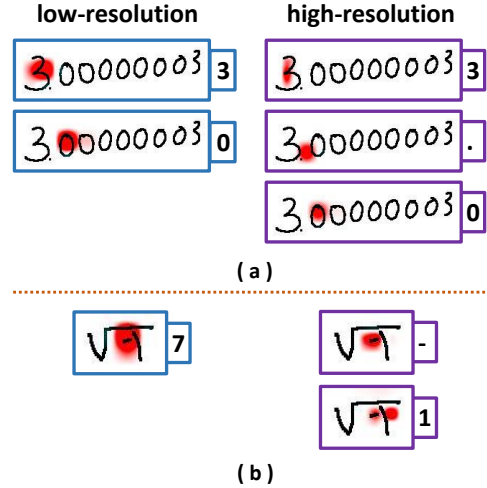

Fig. 3. Two examples of attention visualization on low-resolution annotations and on high-resolution annotations. The attention probabilities are shown through red color and the predicted symbols are shown on the right of images.

TABLE II
COMPARISON OF RECOGNITION PERFORMANCE (IN %) ON CROHME 2014 AND CROHME 2016 WHEN INCREASING THE DEPTH OF DENSE BLOCK IN MULTI-SCALE BRANCH.

Depth	CROHME 2014		CROHME 2016	
	WER	ExpRate	WER	ExpRate
0	13.5	50.8	14.3	48.5
8	13.3	51.3	13.9	49.3
16	12.9	52.8	13.7	50.1
24	13.1	51.4	14.1	48.9

We also illustrate the performance of multi-scale attention model in Fig. 3. The left part of Fig. 3 denotes the visualization of single-scale attention on low-resolution annotations and the right part denotes the visualization of multi-scale attention only on high-resolution annotations. Fig. 3 (a) is an example that the decimal point "." is under-parsed by only relying on low-resolution attention model. However, the high-resolution attention in the multi-scale attention model successfully detects the decimal point. Fig. 3 (b) is an example that the math symbols "- 1" are mis-parsed as "7" due to the low-resolution attention model while the high-resolution attention model can correctly recognize them.

D. Comparison with state-of-the-arts (Q3)

The comparison among the proposed approach and others on CROHME 2014 test set is listed in Table III. Systems I to VII were submitted systems to CROHME 2014 competition and they were mostly based on the traditional two-dimensional context free grammar method. Details of these systems can refer to [27]. To make a fair comparison we erase the system III namely "MyScript" which achieved a high ExpRate of 62.7% but used a large private dataset and the technical details were unrevealed. System "WAP", "CRNN" and our proposed system are all based on encoder-decoder model with attention that takes handwritten mathematical expressions input as images. As for the system "CRNN", it is declared in [16] that the encoder employs a CNN+RNN architecture and the decoder is a unidirectional RNN with classic attention model. Meanwhile a novel data augmentation method for handwritten mathematical expression recognition was proposed in [16]. We can see that our proposed system achieves the best result with ExpRate of 52.8% on CROHME 2014. Additionally, a gap existed between the correct and error percentages ($\leq 1\%$), showing that the corresponding systems have a large room for further improvements. In contrast, the small differences between error ($\leq 2\%$) and error ($\leq 3\%$) illustrate that it is difficult to improve the ExpRate by incorporating a single correction.

TABLE III
COMPARISON OF EXPRATE (IN %) ON CROHME 2014, WE ERASE SYSTEM III NAMELY MYSCRIPT BECAUSE IT USED EXTRA TRAINING DATA.

System	Correct(%)	$\leq 1(\%)$	$\leq 2(\%)$	$\leq 3(\%)$
I	37.2	44.2	47.3	50.2
II	15.0	22.3	26.6	27.7
IV	19.0	28.2	32.4	33.4
V	19.0	26.4	30.8	33.0
VI	25.7	33.2	35.9	37.3
VII	26.1	33.9	38.5	40.0
WAP	**44.4**	**58.4**	**62.2**	**63.1**
CRNN	**35.2**	-	-	-
Ours	**52.8**	**68.1**	**72.0**	**72.7**

TABLE IV
COMPARISON OF EXPRATE (IN %) ON CROHME 2016, WE ERASE TEAM MYSCRIPT BECAUSE IT USED EXTRA TRAINING DATA.

	Correct(%)	$\leq 1 (\%)$	$\leq 2 (\%)$	$\leq 3 (\%)$
Wiris	49.6	60.4	64.7	–
Tokyo	43.9	50.9	53.7	–
São Paolo	33.4	43.5	49.2	–
Nantes	13.3	21.0	28.3	–
WAP	**42.0**	**55.1**	**59.3**	**60.2**
Ours	**50.1**	**63.8**	**67.4**	**68.5**

2249

To complement a more recent algorithm comparison and test the generalization capability of our proposed approach, we also validate our best system on CROHME 2016 test set as shown in Table IV, with an ExpRate of 50.1% which is quite a promising result compared with other participating systems. The system "Wiris" was awarded as the first place on CROHME 2016 competition using only the CROHME training data with an ExpRate of 49.6%, and it used a Wikipedia formula corpus, consisting of more than 592,000 LaTeX notations of mathematical expressions, to train a strong language model. The details of other systems can be found in [28].

V. CONCLUSION

In this study we improve the performance of attention based encoder-decoder for handwritten mathematical expression by introducing the dense encoder and multi-scale attention model. It is the first work that employs densely connected convolutional networks for handwritten mathematical expression recognition and we propose the novel multi-scale attention model to alleviate the problem causing by pooling operation. We demonstrate through attention visualization and experiment results that the novel multi-scale attention model with dense encoder performs better than the state-of-the-art methods.

ACKNOWLEDGMENT

REFERENCES

[1] A. Belaid and J.-P. Haton, "A syntactic approach for handwritten mathematical formula recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 1, pp. 105–111, 1984.

[2] K.-F. Chan and D.-Y. Yeung, "Mathematical expression recognition: a survey," *International Journal on Document Analysis and Recognition*, vol. 3, no. 1, pp. 3–15, 2000.

[3] R. Zanibbi, D. Blostein, and J. R. Cordy, "Recognizing mathematical expressions using tree transformation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 11, pp. 1455–1467, 2002.

[4] F. Álvaro, J.-A. Sánchez, and J.-M. Benedí, "An integrated grammar-based approach for mathematical expression recognition," *Pattern Recognition*, vol. 51, pp. 135–147, 2016.

[5] P. A. Chou, "Recognition of equations using a two-dimensional stochastic context-free grammar," in *Visual Communications and Image Processing IV*, vol. 1199. International Society for Optics and Photonics, 1989, pp. 852–866.

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

[7] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[8] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[9] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4945–4949.

[10] J. Zhang, Y. Zhu, J. Du, and L. Dai, "Radical analysis network for zero-shot learning in printed Chinese character recognition," *arXiv preprint arXiv:1711.01889*, 2017.

[11] ——, "Trajectory-based radical analysis network for online handwritten Chinese character recognition," *arXiv preprint arXiv:1801.10109*, 2018.

[12] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International Conference on Machine Learning*, 2015, pp. 2048–2057.

[13] J. Zhang, J. Du, S. Zhang, D. Liu, Y. Hu, J. Hu, S. Wei, and L. Dai, "Watch, attend and parse: An end-to-end neural network based approach to handwritten mathematical expression recognition," *Pattern Recognition*, 2017.

[14] J. Zhang, J. Du, and L. Dai, "A gru-based encoder-decoder approach with attention for online handwritten mathematical expression recognition," in *Document Analysis and Recognition (ICDAR), 2017 14th International Conference on*. IEEE, 2017.

[15] Y. Deng, A. Kanervisto, and A. M. Rush, "What you get is what you see: A visual markup decompiler," *arXiv preprint arXiv:1609.04938*, 2016.

[16] A. L. Duc and M. Nakagawa, "Training an end-to-end system for handwritten mathematical expression recognition by generated patterns," in *Document Analysis and Recognition (ICDAR), 2017 14th International Conference on*. IEEE, 2017.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[19] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 2013, pp. 6645–6649.

[20] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[21] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," *arXiv preprint arXiv:1608.06993*, 2016.

[22] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[23] J. Zhang, J. Tang, and L.-R. Dai, "Rnn-blstm based multi-pitch estimation." in *INTERSPEECH*, 2016, pp. 1785–1789.

[24] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[25] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315–323.

[26] Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li, "Modeling coverage for neural machine translation," *arXiv preprint arXiv:1601.04811*, 2016.

[27] H. Mouchere, C. Viard-Gaudin, R. Zanibbi, and U. Garain, "ICFHR 2014 competition on recognition of on-line handwritten mathematical expressions (CROHME 2014)," in *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*. IEEE, 2014, pp. 791–796.

[28] H. Mouchère, C. Viard-Gaudin, R. Zanibbi, and U. Garain, "ICFHR2016 CROHME: Competition on recognition of online handwritten mathematical expressions," in *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*. IEEE, 2016, pp. 607–612.

[29] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[30] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[31] K. Cho, "Natural language understanding with distributed representation," *arXiv preprint arXiv:1511.07916*, 2015.

[32] T. G. Dietterich *et al.*, "Ensemble methods in machine learning," *Multiple classifier systems*, vol. 1857, pp. 1–15, 2000.

2250