A GRU-based Encoder-Decoder Approach with Attention for Online 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*—In this study, we present a novel end-to-end approach based on the encoder-decoder framework with the attention mechanism for online handwritten mathematical expression recognition (OHMER). First, the input two-dimensional ink trajectory information of handwritten expression is encoded via the gated recurrent unit based recurrent neural network (GRU-RNN). Then the decoder is also implemented by the GRU-RNN with a coverage-based attention model. The proposed approach can simultaneously accomplish the symbol recognition and structural analysis to output a character sequence in LaTeX format. Validated on the CROHME 2014 competition task, our approach significantly outperforms the state-of-the-art with an expression recognition accuracy of 52.43% by only using the official training dataset. Furthermore, the alignments between the input trajectories of handwritten expressions and the output LaTeX sequences are visualized by the attention mechanism to show the effectiveness of the proposed method.**

***Keywords*-Online Handwritten Mathematical Expression Recognition, Encoder-Decoder, Gated Recurrent Unit, Attention**

I. Introduction

Mathematical expressions are indispensable for describing problems and theories in math, physics and many other fields. With the rapid development of pen-based interfaces and tactile devices, people are allowed to write mathematical expressions on mobile devices using handwriting. However, the automatic recognition of these handwritten mathematical expressions is quite different from the traditional character recognition problems with more challenges [1]–[3], e.g., the complicated geometric structures, enormous ambiguities in handwritten input and the strong dependency on contextual information. This study focuses on the online handwritten mathematical expression recognition (OHMER), which attracts broad attention such as the Competition on Recognition of Online Handwritten Mathematical Expressions (CROHME) [4].

OHMER consists of two major problems [5], [6], namely symbol recognition and structural analysis, which can be solved sequentially or globally. In the sequential solutions [7], [8], the errors of symbol recognition and segmentation are subsequently inherited by the structural analysis. Consequently, the global solutions [9], [10] can well address this problem, which are computationally expensive as the probabilities for segmentation composed of strokes are exponentially expanded. Many approaches for structural analysis of mathematical expressions have been investigated, including expression trees [11], two-dimensional hidden Markov model (HMM) [12] and others [13]–[15]. Among these, the grammar-based methods [16], [17] are widely used in OHMER systems [8]–[10]. These grammars are constructed using extensive prior knowledge with the corresponding parsing algorithms. Overall, both conventional sequential and global approaches have common limitations: 1) the challenging symbol segmentation should be explicitly designed; 2) structural analysis requires the priori knowledge or rules; 3) the computational complexity of parsing algorithms increases exponentially with the size of the predefined grammar.

To address these problems, in this paper, we propose a novel end-to-end approach using the attention based encoder-decoder model with recurrent neural networks (RNNs) [18] for OHMER. First, the input two-dimensional ink trajectory information of handwritten expression is encoded to the high-level representations via the stack of bi-directional gated recurrent unit based recurrent neural network (GRU-RNN). Then the decoder is implemented by a unidirectional GRU-RNN with a coverage-based attention model [19]–[21]. The attention mechanism built into the decoder scans the entire input sequence and chooses the most relevant region to describe a segmented symbol or implicit spatial operator. Inherently unlike traditional approaches, our model optimizes symbol segmentation automatically through its attention mechanism, and structural analysis does not rely on a predefined grammar. Moreover, the encoder and the decoder are jointly trained. The proposed encoder-decoder architecture [22] can make the symbol segmentation, symbol recognition, and structural analysis unified in one data-driven framework to output a character sequence in LaTeX format [23]. Validated on the CROHME 2014 competition task, our approach significantly outperforms the state-of-the-art with an expression recognition accuracy of 52.43% by only using the official training dataset. Furthermore, the alignments between the input trajectories of handwritten expressions and the output LaTeX sequences are visualized by the attention mechanism to show the effectiveness of the proposed method.

Our proposed approach is related to our previous work [24] and a recent work [25] with the new contributions as: 1) the

2379-2140/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDAR.2017.152

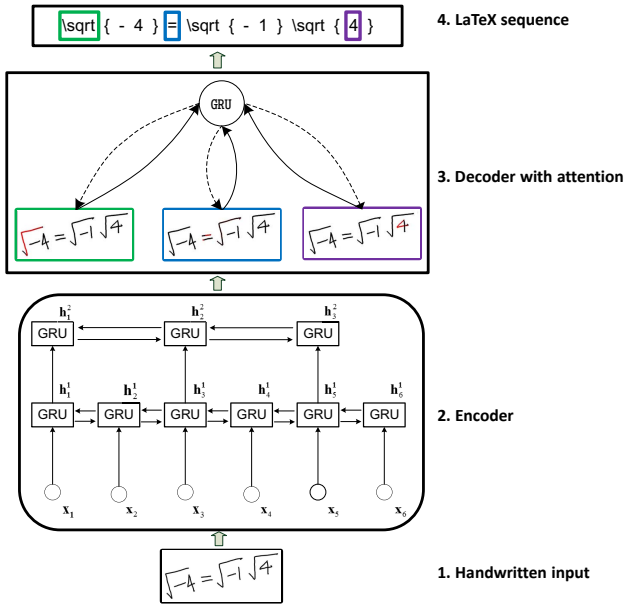

Fig. 1. The overall architecture of attention based encoder-decoder model.

encoder in this work can fully utilize the online trajectory information via the GRU-RNN while the encoder in [24] using convolutional neural network (CNN) can only work for the offline image as input; 2) different from [25], the newly added coverage-based attention model is crucial to the recognition performance and its visualization can well explain the effectiveness of the proposed method.

The remainder of the paper is organized as follows. In Section II, the details of the proposed approach are introduced. In Section III, the experimental results and analysis are reported. Finally the conclusion is given in Section IV.

II. The Proposed Approach

In this section, we elaborate the proposed end-to-end framework, namely generating an underlying LaTeX sequence from a sequence of online handwritten trajectory points, as illustrated in Fig. 1. First, the preprocessing is applied to the original trajectory points to extract the input feature vector. Then, the encoder and decoder are well designed using the GRU-RNNs [26]. The encoder is a stack of bidirectional GRUs while the decoder combines a unidirectional GRU and an attention mechanism into the recurrent sequence generator. The attention mechanism can potentially well learn the alignment between the input trajectory and the output LaTeX sequence. For example in Fig. 1, the green, blue, and purple rectangles denote three symbols with the red color representing the attention probabilities of each handwritten symbol.

A. Preprocessing

Suppose the input handwritten mathematical expression consists of a sequence of trajectory points with a variable-length N:

$$\{[x_1, y_1, s_1],\ [x_2, y_2, s_2],\ \ldots,\ [x_N, y_N, s_N]\} \tag{1}$$

where x_i and y_i are the xy-coordinates of the pen movements and s_i indicates which stroke the i^{th} point belongs to.

To address the issue of non-uniform sampling by different writing speed and the size variations of the coordinates on different potable devices, the interpolation and normalization to the original trajectory points are first conducted according to [27]. Then we extract an 8-dimensional feature vector for each point:

$$\left[x_i, y_i, \Delta x_i, \Delta y_i, \Delta^2 x_i, \Delta^2 y_i, \delta(s_i = s_{i+1}), \delta(s_i \neq s_{i+1})\right] \tag{2}$$

where $\Delta x_i = x_{i+1} - x_i$, $\Delta y_i = y_{i+1} - y_i$, $\Delta^2 x_i = x_{i+2} - x_i$, $\Delta^2 y_i = y_{i+2} - y_i$ and $\delta(\cdot) = 1$ when the condition is true or zero otherwise. The last two terms are flags which indicate the status of the pen, i.e., $[1,\ 0]$ and $[0,\ 1]$ are pen-down and pen-up, respectively. A handwritten mathematical expression is actually composed of several strokes. So fully utilizing the stroke segmentation information plays an important role in constructing an effective recognizer. For convenience, in the following sections, we use $\mathbf{X} = (\mathbf{x}_1,\ \mathbf{x}_2,\ \ldots,\ \mathbf{x}_N)$ to denote the input sequence of the encoder, where $\mathbf{x}_i \in \mathbb{R}^d$ $(d = 8)$.

B. Encoder

Given the input sequence $(\mathbf{x}_1,\ \mathbf{x}_2,\ \ldots,\ \mathbf{x}_N)$, a simple RNN can be adopted as an encoder to compute the corresponding sequence of hidden state $(\mathbf{h}_1,\ \mathbf{h}_2,\ \ldots,\ \mathbf{h}_N)$:

$$\mathbf{h}_t = \tanh\left(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1}\right) \tag{3}$$

where $\mathbf{W}_{xh}$ is the connection weight matrix of the network between input layer and hidden layer, and $\mathbf{W}_{hh}$ is the weight matrix of recurrent connections in a hidden layer. In principle, the recurrent connections can make RNN map from the entire history of previous inputs to each output. However, in practice, a simple RNN is difficult to train properly due to the problems of the vanishing gradient and the exploding gradient as described in [28], [29].

Therefore, in this study, we utilize GRU as an improved version of simple RNN which can alleviate the vanishing and exploding gradient problem. The encoder GRU hidden state $\mathbf{h}_t$ is computed as follows:

$$\mathbf{z}_t = \sigma(\mathbf{W}_{xz}\mathbf{x}_t + \mathbf{U}_{hz}\mathbf{h}_{t-1}) \tag{4}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_{xr}\mathbf{x}_t + \mathbf{U}_{hr}\mathbf{h}_{t-1}) \tag{5}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{U}_{rh}(\mathbf{r}_t \otimes \mathbf{h}_{t-1})) \tag{6}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \otimes \mathbf{h}_{t-1} + \mathbf{z}_t \otimes \tilde{\mathbf{h}}_t \tag{7}$$

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. $\mathbf{W}_{xz}$, $\mathbf{W}_{xr}$, $\mathbf{W}_{xh}$, $\mathbf{U}_{hz}$, $\mathbf{U}_{hr}$ and $\mathbf{U}_{rh}$ are related weight matrices.

Nevertheless, unidirectional GRU cannot utilize the future context. To address this issue, we pass the input vectors through two GRU layers running in opposite directions and

concatenate their hidden state vectors. This bidirectional GRU can use both past and future information. To obtain a better representation for the decoder to attend, we stack multiple layers of GRU on top of each other as the encoder. However, as the depth of encoder increases, the high-level representation might contain much redundant information. So we add pooling over time in high-level GRU layers as illustrated by Fig. 1. The pooling operation not only helps accelerate the training process, but also improves the recognition performance as the decoder is easier to attend with a fewer number of outputs of encoder.

C. Decoder equipped with attention mechanism

As shown in Fig. 1, the decoder generates a corresponding LaTeX sequence of the input handwritten mathematical expression. The output sequence $\mathbf{Y}$ is encoded as a sequence of one-shot vectors.

$$\mathbf{Y} = \{\mathbf{y_1}, \ldots, \mathbf{y_C}\} \ , \ \mathbf{y_i} \in \mathbb{R}^{\mathbf{K}} \tag{8}$$

where K is the number of total symbols/words in the vocabulary and C is the length of a LaTeX sequence. Meanwhile, the bi-directional GRU encoder produces an annotation sequence $\mathbf{A}$ with a length L. If there is no pooling in the bi-directional GRU encoder, $L = N$. Each of these annotations is a D-dimensional vector:

$$\mathbf{A} = \{\mathbf{a_1}, \ldots, \mathbf{a_L}\} \ , \ \mathbf{a_i} \in \mathbb{R}^{\mathbf{D}} \tag{9}$$

Note that, both the length of annotation sequence L and the length of LaTeX sequence C are not fixed. 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-size vector $\mathbf{c}_t$, which will be described later. Given the context vector $\mathbf{c}_t$, we utilize unidirectional GRU to produce the LaTeX sequences symbol by symbol. The probability of each predicted symbol is calculated as:

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

where g denotes a softmax activation function over all the symbols in the vocabulary. $\mathbf{s}_t$ is the current hidden state of the GRU decoder and $\mathbf{y}_{t-1}$ represents the previous target symbol. $\mathbf{W}_o \in \mathbb{R}^{K\times m}$, $\mathbf{W}_s \in \mathbb{R}^{m\times n}$, $\mathbf{W}_c \in \mathbb{R}^{m\times D}$, and $\mathbf{E}$ denotes the embedding matrix. m and n are the dimensions of embedding and GRU decoder. The GRU decoder also takes the previous target symbol $\mathbf{y}_{t-1}$ and the context vector $\mathbf{c}_t$ as input, and employs a single unidirectional GRU layer to calculate the hidden state $\mathbf{s}_t$:

$$\mathbf{z}'_t = \sigma(\mathbf{W}_{yz}\mathbf{E}\mathbf{y}_{t-1} + \mathbf{U}_{sz}\mathbf{s}_{t-1} + \mathbf{C}_{cz}\mathbf{c}_t) \tag{11}$$

$$\mathbf{r}'_t = \sigma(\mathbf{W}_{yr}\mathbf{E}\mathbf{y}_{t-1} + \mathbf{U}_{sr}\mathbf{s}_{t-1} + \mathbf{C}_{cr}\mathbf{c}_t) \tag{12}$$

$$\tilde{\mathbf{s}}_t = \tanh(\mathbf{W}_{ys}\mathbf{E}\mathbf{y}_{t-1} + \mathbf{U}_{rs}(\mathbf{r}'_t \otimes \mathbf{s}_{t-1}) + \mathbf{C}_{cs}\mathbf{c}_t) \tag{13}$$

$$\mathbf{s}_t = (1 - \mathbf{z}'_t) \otimes \mathbf{s}_{t-1} + \mathbf{z}'_t \otimes \tilde{\mathbf{s}}_t \tag{14}$$

where $\mathbf{z}'_t$, $\mathbf{r}'_t$ and $\tilde{\mathbf{s}}_t$ are the update gate, reset gate and candidate activation, respectively. $\mathbf{W}_{yz}$, $\mathbf{W}_{yr}$, $\mathbf{W}_{ys}$, $\mathbf{U}_{sz}$, $\mathbf{U}_{sr}$, $\mathbf{U}_{rs}$, $\mathbf{C}_{cz}$, $\mathbf{C}_{cr}$ and $\mathbf{C}_{cs}$ are related weight matrices.

Intuitively, for each predicted symbol from the decoder, not the entire input sequence is useful. Only a subset of adjacent trajectory points should mainly contribute to the computation of context vector $\mathbf{c}_t$ at each time step t. Therefore, the decoder can adopt an attention mechanism to link to the related part of input sequence and then assign a higher weight to the corresponding annotation vector $\mathbf{a}_i$. Here, we parameterize the attention model as a multi-layer perceptron (MLP) that is jointly trained with the encoder and the decoder:

$$e_{ti} = \boldsymbol{\nu}_{att}^{\mathrm{T}} \tanh(\mathbf{W}_{att}\mathbf{s}_{t-1} + \mathbf{U}_{att}\mathbf{a}_i) \tag{15}$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{L}\exp(e_{tk})} \tag{16}$$

Let n' denote the attention dimension. Then $\boldsymbol{\nu}_{att} \in \mathbb{R}^{n'}$, $\mathbf{W}_{att} \in \mathbb{R}^{n'\times n}$ and $\mathbf{U}_a \in \mathbb{R}^{n'\times D}$. With the weights α_{ti}, the context vector $\mathbf{c}_t$ is calculated as:

$$\mathbf{c}_t = \sum\nolimits_i^L \alpha_{ti}\mathbf{a}_i \tag{17}$$

The attention probability α_{ti} denotes the alignment between the target symbol and a local region of input sequence. It can also be considered as a regularization parameter for the bi-directional GRU encoder because the attention helps diminish the gradient back-propagated from the decoder.

D. Coverage based attention model

However, there is one problem for the conventional attention mechanism in (15), namely the lack of coverage [30]. Coverage represents overall alignment information. An attention model lacking coverage is not aware whether a part of input expression has been translated or not. Misalignment will lead to over-translating or under-translating. Over-translating means that some parts of the input sequence have been translated twice or more, while under-translating implies that some parts have never been translated. To address this problem, we append a coverage vector to the computation of attention in (15). The coverage vector aims at providing alignment information. Different from [31], we compute the coverage vector based on the sum of all past attention probabilities $\boldsymbol{\beta}_t$, which can describe the alignment history:

$$\boldsymbol{\beta}_t = \sum\nolimits_l^{t-1} \boldsymbol{\alpha}_l \tag{18}$$

$$\mathbf{F} = \mathbf{Q} * \boldsymbol{\beta}_t \tag{19}$$

$$e_{ti} = \boldsymbol{\nu}_{att}^{\mathrm{T}} \tanh(\mathbf{W}_{att}\mathbf{s}_{t-1} + \mathbf{U}_{att}\mathbf{a}_i + \mathbf{U}_f\mathbf{f}_i) \tag{20}$$

where $\boldsymbol{\alpha}_l$ is the attention probability vector at time step l and $\mathbf{f}_i$ denotes the i^{th} coverage vector of $\mathbf{F}$. $\boldsymbol{\beta}_t$ is initialized as a zero vector. The coverage vector is produced through a convolutional layer because we believe the coverage vector of annotation $\mathbf{a}_i$ should also be associated with its adjacent attention probabilities.

The coverage vector is expected to adjust the future attention. More specifically, trajectory points in the input sequence already significantly contributed to the generation of target symbols should be assigned with lower attention probabilities

in the following decoding phases. On the contrary, trajectory points with less contributions should be assigned with higher attention probabilities. Consequently, the decoding process is finished only when the entire input sequence has contributed and the problems of over-translating or under-translating can be alleviated.

III. Experiments

The experiments are conducted on CROHME 2014 competition dataset. The training set consists of 8836 handwritten mathematical expressions (about 86000 symbols) while the test set includes 986 expressions (about 6000 symbols). There are totally 101 maths symbol classes. None of the expressions in the test set is seen in the training set. To be fairly comparable, we also used the CROHME 2013 test set as a validation set in the training stage, just like other participants of CROHME 2014 competition.

The training objective of our model is to maximize the predicted symbol probability as shown in (10) and we use cross-entropy (CE) as the criterion. The encoder consists of 4 layers of bi-directional GRUs. Each layer has 250 forward and 250 backward GRU units. The pooling is applied to the top 2 GRU layers over time. Accordingly, the encoder reduces the input sequence length by the factor of 4. The decoder is a single layer with 256 forward GRU units. The embedding dimension m and GRU decoder dimension n are set to 256. The attention dimension n' and annotation dimension D are set to 500. We utilize the AdaDelta algorithm [32] with gradient clipping for optimization. The AdaDelta hyperparameters are set as $\rho = 0.95$, $\varepsilon = 10^{-6}$. The early-stopping of training procedure is determined by word error rate (WER) [33] of validation set. We use the weight noise [34] as the regularization. The training is first finished without weight noise, we then anneal the best model in terms of WER by restarting the training with weight noise.

In the decoding stage, we aim to generate a most likely LaTeX sequence given the input sequence. The beam search algorithm [35] is employed to complete the decoding process. At each time step, we maintain a set of 10 partial hypotheses. We also adopt the ensemble method [36] to improve the performance of our neural network model. We first train 5 models on the same training set but with different initialized parameters and then average their prediction probabilities on the generated symbol during the beam search process.

A. Recognition performance

The comparison among the proposed approach (systems P1, P2, P3) and others on CROHME 2014 test set is listed in Table I. Systems I to VII were submitted systems to CROHME 2014 competition. Note that system III is not given for a fair comparison as it used additional training data not provided officially. The details of these systems can be seen in [4]. System I acquired an expression rate (ExpRate) of 37.22% and was awarded the first place on CROHME 2014 competition using the official training data. It should be indicated that there is a large performance gap between the ExpRate of the first place and the second place.

TABLE I
Correct expression recognition rates (in %) of different systems on CROHME 2014 test set.

System	Correct(%)	$\leq$ 1(%)	$\leq$ 2(%)	$\leq$ 3(%)
I	37.22	44.22	47.26	50.20
II	15.01	22.31	26.57	27.69
IV	18.97	28.19	32.35	33.37
V	18.97	26.37	30.83	32.96
VI	25.66	33.16	35.90	37.32
VII	26.06	33.87	38.54	39.96
P1	**42.49**	**57.91**	**60.45**	**61.56**
P2	**46.86**	**61.87**	**65.82**	**66.63**
P3	**52.43**	**68.05**	**71.50**	**72.31**

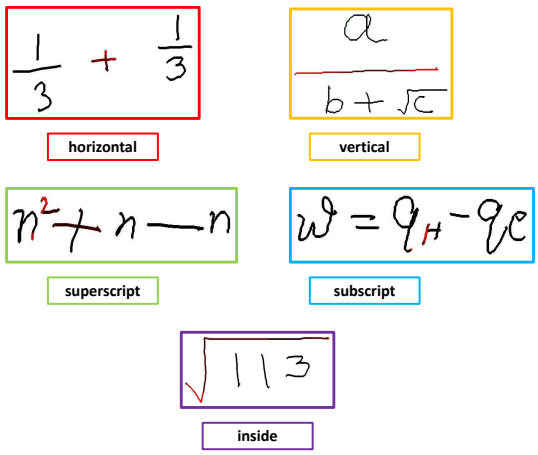

Fig. 2. The learning of five spatial relationships (horizontal, vertical, subscript, superscript and inside) through attention visualization.

System P1 and P2 are two of our proposed systems without/with coverage based attention model, respectively. System P3 is our best ensemble system with 5 models. It is clear that system P1 even without coverage model can achieve an ExpRate of 42.49%, which significantly outperforms the best submitted system to CROHME 2014 competition with an absolute gain of about 5%. By using the coverage model, an absolute gain of 4% could be obtained from system P1 to P2. The best system P3 yields an ExpRate of 52.43%, which should be the best published result on CROHME 2014 test set, to the best of our knowledge.

A mathematical expression is considered to be correctly recognized only when the generated LaTeX sequence matches ground truth. Additionally, Table I also shows the expression recognition accuracies with one, two and three errors per expression, represented by ($\leq$ 1), ($\leq$ 2) and ($\leq$ 3). The performance gap between correct and error ($\leq$ 1) shows that the corresponding systems still have a large room to be improved. Meanwhile, the differences between error (≤ 2) and error (≤ 3) show that it is difficult to improve the accuracy by incorporating a single correction when more errors happen.

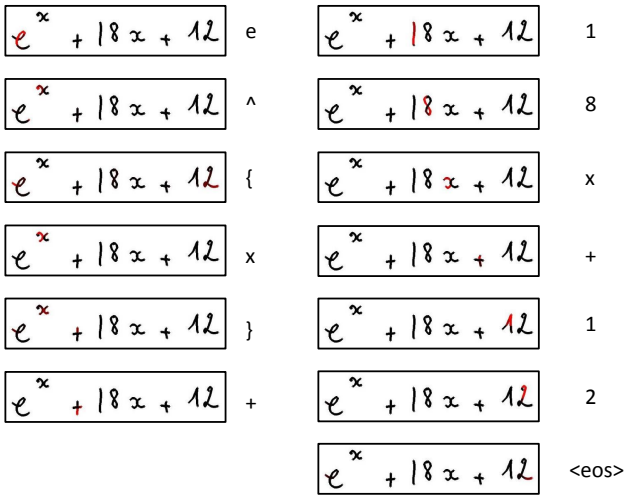

Fig. 3. Attention visualization for an example of a handwriting mathematical expression with the LaTeX ground truth " e ∧ { x } + 1 8 x + 1 2 ".

B. Attention visualization

In this section, we show through attention visualization how the proposed model is able to analyse the two-dimensional structure grammar and perform symbol segmentation implicitly. We draw the trajectory of input handwritten mathematical expression in a 2-D image to visualize attention. We use the red color to describe the attention probabilities, namely the higher attention probabilities with the lighter color and the lower attention probabilities with the darker color.

In the two-dimensional grammar of mathematical expressions, there are mainly five kinds of spatial relationship between maths symbols, including horizontal, vertical, subscript, superscript and inside relationships. Correctly recognizing these five spatial relationships is the key to analyse the two-dimensional grammar. As shown in Fig. 2, the horizontal and vertical relationships are easy to learn by focusing on the middle operator. When dealing with superscripts, the decoder precisely pays attention to the end of base symbols and the start of superscript symbols. It does make sense because trajectory points in the start of superscript symbols are on the upper-right of trajectory points in the end of base symbols, describing the upper-right direction. Similarly, for subscripts, the ending points of base symbols and the starting points of superscript symbols can also describe the bottom-right direction. As for the inside relationships, the decoder attends to the bounding symbols.

More specifically, in Fig. 3, we take the expression $e^x + 18x + 12$ as a correctly recognized example. We show that how our model learns to translate this handwritten mathematical expression from a sequence of trajectory points into a LaTeX sequence " e ∧ { x } + 1 8 x + 1 2 " step by step. When encountering basic math symbols like "e", "x", "+", "1", "2" and "8", the attention model well generates the alignment strongly corresponding to the human intuition. When encountering a spatial relationship in e^x, the attention model correctly

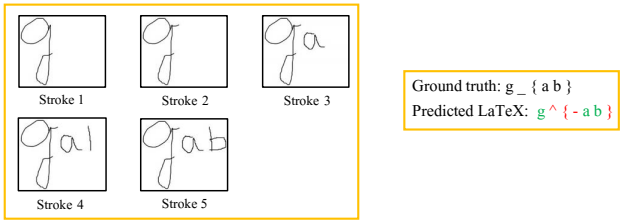

Fig. 4. Analysis of an incorrectly recognized example of handwritten mathematical expression due to the over-translating problem where "∧" is over-translated.

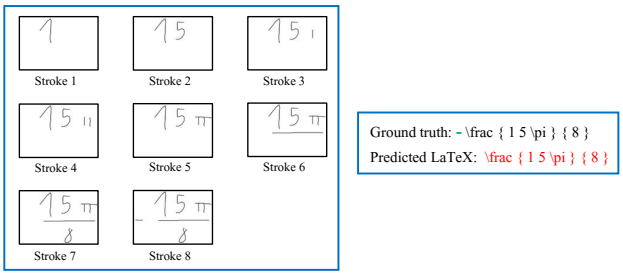

Fig. 5. Analysis of an incorrectly recognized example of handwritten mathematical expression due to the under-translating problem where the minus sign "–" is under-translated.

distinguishes the upper-right direction and then produces the symbol "∧". More interestingly, immediately after detecting the superscript spatial relationship, the decoder successfully generates a pair of braces "{}", which are used to compose the exponent grammar in LaTeX. Finally, the decoder attends both the end and the start of the entire input sequence and generates an end-of-sentence (eos) mark.

C. Error Analysis

In Fig. 4 and Fig. 5, we show two typical incorrectly recognized examples of handwritten mathematical expressions, due to over-translating and under-translating, respectively. The stroke 2 in Fig. 4 is an inserted stroke, which is actually the end of symbol "g" but split into another stroke by the writer. Accordingly, our model over-translates the input sequence and recognizes the stroke 2 as a minus sign "–". And the spatial relationship subscript is mistaken as the superscript. In Fig. 5, the first symbol of formula LaTeX string, namely the minus sign "–", is missing, which corresponds to the last stroke of the handwritten example. In general, we should write the minus sign as the first stroke. Consequently, this inverse stroke leads to the under-translating problem.

IV. Conclusion

In this study we introduce an encoder-decoder with coverage based attention model to recognize online handwritten mathematical expressions. The proposed approach can fully utilize the online trajectory information via GRU-RNN based encoder. And the coverage model is quite effective for attention by using the alignment history information. We achieve

promising recognition results on CROHME 2014 competition. We show from experiment results that our model is capable of performing symbol segmentation automatically and learning to grasp a maths grammar without priori knowledge. Also, we demonstrate through attention visualization that the learned alignments by attention model well correspond to human intuition. As for the future work, we aim to improve our approach to reduce the over-translating and under-translating errors.

ACKNOWLEDGMENT

The authors want to thank Junfeng Hou for insightful comments and suggestion. This work was supported in part by the National Natural Science Foundation of China under Grants 61671422 and U1613211, in part by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDB02070006, and in part by the National Key Research and Development Program of China under Grant 2016YFB1001300.

REFERENCES

[1] R. H. Anderson, "Syntax-directed recognition of hand-printed two-dimensional mathematics," in *Symposium on Interactive Systems for Experimental Applied Mathematics: Proceedings of the Association for Computing Machinery Inc. Symposium*. ACM, 1967, pp. 436–459.

[2] 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.

[3] E. G. Miller and P. A. Viola, "Ambiguity and constraint in mathematical expression recognition," in *AAAI/IAAI*, 1998, pp. 784–791.

[4] 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.

[5] 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.

[6] R. Zanibbi and D. Blostein, "Recognition and retrieval of mathematical expressions," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 15, no. 4, pp. 331–357, 2012.

[7] 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.

[8] F. Álvaro, J.-A. Sánchez, and J.-M. Benedí, "Recognition of on-line handwritten mathematical expressions using 2D stochastic context-free grammars and hidden Markov models," *Pattern Recognition Letters*, vol. 35, pp. 58–67, 2014.

[9] A.-M. Awal, H. Mouchère, and C. Viard-Gaudin, "A global learning approach for an online handwritten mathematical expression recognition system," *Pattern Recognition Letters*, vol. 35, pp. 68–77, 2014.

[10] 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.

[11] J. Ha, R. M. Haralick, and I. T. Phillips, "Understanding mathematical expressions from document images," in *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, vol. 2. IEEE, 1995, pp. 956–959.

[12] A. Kosmala and G. Rigoll, "On-line handwritten formula recognition using statistical methods," in *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*, vol. 2. IEEE, 1998, pp. 1306–1308.

[13] H.-J. Lee and M.-C. Lee, "Understanding mathematical expressions in a printed document," in *Document Analysis and Recognition, 1993., Proceedings of the Second International Conference on*. IEEE, 1993, pp. 502–505.

[14] H.-J. Winkler, H. Fahrner, and M. Lang, "A soft-decision approach for structural analysis of handwritten mathematical expressions," in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 4. IEEE, 1995, pp. 2459–2462.

[15] N. S. Hirata and F. D. Julca-Aguilar, "Matching based ground-truth annotation for online handwritten mathematical expressions," *Pattern Recognition*, vol. 48, no. 3, pp. 837–848, 2015.

[16] P. A. Chou, "Recognition of equations using a two-dimensional s-tochastic context-free grammar," *Visual Communications and Image Processing IV*, vol. 1199, pp. 852–863, 1989.

[17] S. Lavirotte and L. Pottier, "Mathematical formula recognition using graph grammar," in *Photonics West'98 Electronic Imaging*. International Society for Optics and Photonics, 1998, pp. 44–52.

[18] A. Graves, "Supervised sequence labelling with recurrent neural networks."

[19] 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.

[20] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent nn: First results," *arXiv preprint arXiv:1412.1602*, 2014.

[21] 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.

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

[23] L. Lamport, *LaTeX: A document preparation system: User's guide and reference. illustrations by Duane Bibby*. Reading, Mass: Addison-Wesley Professional. ISBN 0-201-52983-1, 1994.

[24] 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.

[25] 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.

[26] 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.

[27] X.-Y. Zhang, F. Yin, Y.-M. Zhang, C.-L. Liu, and Y. Bengio, "Drawing and recognizing Chinese characters with recurrent neural network," *arXiv preprint arXiv:1606.06539*, 2016.

[28] 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.

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

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

[31] 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.

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

[33] D. Klakow and J. Peters, "Testing the correlation of word error rate and perplexity," *Speech Communication*, vol. 38, no. 1, pp. 19–28, 2002.

[34] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems*, 2011, pp. 2348–2356.

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

[36] T. G. Dietterich, "Ensemble methods in machine learning," in *International workshop on multiple classifier systems*. Springer, 2000, pp. 1–15.