2020 25th International Conference on Pattern Recognition (ICPR)
Milan, Italy, Jan 10-15, 2021

Stroke Based Posterior Attention for Online Handwritten Mathematical Expression Recognition

Changjie Wu, Qing Wang, Jianshu Zhang, Jun Du*, Jiaming Wang
NEL-SILP
University of Science and Technology of China
Hefei, Anhui, P. R. China
wucj@mail.ustc.edu.cn, xiaosong@mail.ustc.edu.cn, xysszjs@mail.ustc.edu.cn,
✉jundu@ustc.edu.cn, jmwang66@mail.ustc.edu.cn

Jiajia Wu, Jinshui Hu
AI Research
iFLYTEK CO.,LTD.
Hefei, Anhui, P. R. China
jjwu@iflytek.com, jshu@iflytek.com

Abstract—**Recently, many researches propose to employ attention based encoder-decoder models to convert a sequence of trajectory points into a LaTeX string for online handwritten mathematical expression recognition (OHMER), and the recognition performance of these models critically relies on the accuracy of the attention. In this paper, unlike previous methods which basically employ a soft attention model, we propose to employ a posterior attention model, which modifies the attention probabilities after observing the output probabilities generated by the soft attention model. In order to further improve the posterior attention mechanism, we propose a stroke average pooling layer to aggregate point-level features obtained from the encoder into stroke-level features. We argue that posterior attention is better to be implemented on stroke-level features than point-level features as the output probabilities generated by stroke is more convincing than generated by point, and we prove that through experimental analysis. Validated on the CROHME competition task, we demonstrate that stroke based posterior attention achieves expression recognition rates of 54.26% on CROHME 2014 and 51.75% on CROHME 2016. According to attention visualization analysis, we empirically demonstrate that the posterior attention mechanism can achieve better alignment accuracy than the soft attention mechanism.**

Index Terms—**Posterior attention, Stroke, Online handwritten mathematical expression recognition, Sequence-to-sequence learning;**

I. Introduction

With the development of digital products and the promotion of paperless office, online handwriting input has become a popular input method. Online handwritten mathematical formula plays an indispensable role in digital education, scientific research, online testing and other scenarios. Although many systems for online handwritten text recognition have been mature with the help of deep neural networks, there is still much room for improvement in online handwritten mathematical expression recognition (OHMER) [1], [2]. OHMER aims to convert the coordinates of human handwritten trajectory points into a format file that a computer can process [3], such as LaTeX strings and inkml [4]. Compared with online handwritten text recognition problems [5], OHMER faces two distinctive challenges: complex two-dimensional spatial structure and smaller open datasets.

For OHMER, we not only need to accurately recognize each mathematical symbol, but also recognize the correct structural relationship between the mathematical symbols, such as superscript, subscript, etc. We can generally divide recognition methods into two types by the number of pipelines: two-step methods [6], [7] and end-to-end methods [2], [8], [9].

In two-step methods, the first pipeline is symbol recognition that recognize trajectory points into mathematical symbols and the second pipeline is structure recognition that parses formula structure from given mathematical symbols. Symbol recognition can be achieved by neural networks or traditional methods [10]. Structure recognition can be analyzed by two-dimensional context free grammar [11], [12]. The shortcomings of the step-by-step recognition method are as follows: First, prior knowledge of mathematics is required to write complex algorithmic rules to parse complex mathematical expressions. Second, the structure recognition depends on the result of the symbol recognition, which means it can lead to error accumulation. The end-to-end system performs both symbol recognition and structure recognition in one single pipeline [13]–[16]. These systems are usually based on the encoder-decoder framework [17], [18], which can convert a sequence of trajectory points into a LaTeX string for mathematical expression recognition. One of the characteristics of the end-to-end method is that its performance heavily depends on the alignment information obtained by the attention mechanism.

In this study, we implement the proposed stroke based posterior attention network on the previous state-of-the-art end-to-end model, named Track, Attend and Parse (TAP) [2]. This paper improves the TAP on the following two aspects. First, we replace the soft attention mechanism with the posterior attention mechanism [19]. In posterior attention mechanism, at each decoding step, the soft attention probabilities of each trajectory point are first computed. We then use each point as the input of symbol classification so that we can get the output probabilities of each point. The posterior attention are computed by normalizing the soft attention probabilities of all points and taking the output probabilities as the confidence of each point. Therefore, posterior attention can get better alignment than soft attention as it considers the posterior information of each point. Although TAP proposes an attention guider to enhance the soft attention mechanism, it needs additional labelled alignment data to construct the guider and such alignment data is hard to be labelled. Besides, the posterior attention mechanism can still improve the soft attention even the attention model is trained with guider.

Second, we argue that the posterior attention mechanism has a condition that the output posterior probabilities generated by each point should be trustworthy, otherwise the posterior information will not be accurate. However, for OHMER, one math symbol is usually composed of tens or hundreds of points, which means one point is not enough to describe one math symbol. To solve this problem, we propose to compute posterior attention on strokes other than points as stroke-level features have more concentrated, richer information than point-level features, and are more suitable for the posterior attention mechanism. We aggregate the corresponding local features of all points in the same stroke into the stroke-level feature by using a stroke average pooling layer. We validate the effectiveness of stroke based posterior attention through experiments on the CROHME competition dataset [4], [10].

The main contributions are summarized as follows:

- We propose to use posterior attention mechanism for OHMER, which can significantly improve the attention alignment and the recognition performance than soft attention.
- We aggregate the point-level features into the stroke-level features, which are more suitable for the posterior attention mechanism.
- We do detailed experimental analysis on the CROHME competition dataset to analyze the advantages of posterior attention compared to soft attention.

The rest of this paper is organized as follows. In Section II we illustrate the structure of stroke based posterior attention in detail. In Section III, we present and analyze the experimental results. In Section IV, we conclude this study.

II. METHODOLOGY

In this section, we first introduce the overall framework of the model: the encoder-decoder framework, which takes the trajectory points as input and outputs a latex string. Then we introduce the improvement in the encoder: the stroke average pooling layer, which aggregate the features obtained by the encoder module from the point level to the stroke level. Finally, we introduce the improvement in the decoder: the posterior attention mechanism, which is a statistically more reasonable and accurate attention mechanism.

A. Encoder-Decoder Framework

As shown in Fig. 1, the architecture of our proposed stroke based posterior attention includes two parts: GRU [20] encoder with stroke average pooling and GRU decoder with posterior attention. First, we represent the raw input as a sequence of N_p points.

$$\mathbf{X}_{\text{raw}} = \{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_{N_p}\}, \mathbf{p}_i = [x_i, y_i, s_i] \tag{1}$$

Each point is composed of spatial coordinates (x_i, y_i) and a stroke index s_i. The stroke index s_i indicates which stroke the point belongs to. Then, a pre-processing process [21] is

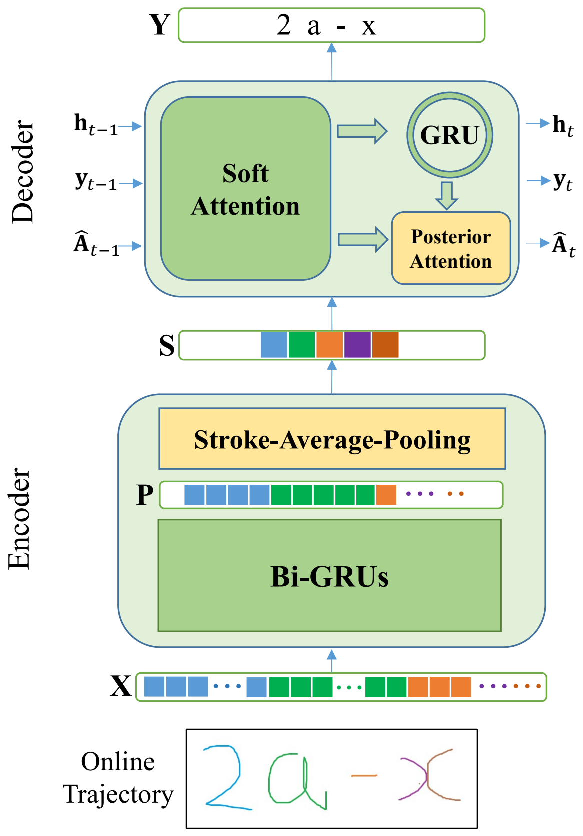

Fig. 1. Architecture of the stroke based posterior attention mechanism based encoder-decoder model.

implemented to extract an 8-dimensional feature vector $\mathbf{x}_i$ from each point.

$$\mathbf{x}_i = [x_i, y_i, \Delta x_i, \Delta y_i, \Delta' x_i, \Delta' y_i, \delta_i, \bar{\delta}_i] \tag{2}$$

$$\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{N_p}\} \tag{3}$$

where $\Delta x_i = x_{i+1} - x_i, \Delta y_i = y_{i+1} - y_i, \Delta' x_i = x_{i+2} - x_i, \Delta' y_i = y_{i+2} - y_i, \delta = 1$ when $s_i = s_{i+1}$ or otherwise 0 and $\bar{\delta}_i = 1 - \delta_i$. We use $\mathbf{X}$ as the input to the encoder. The encoder consists of four Bi-GRU layers, two max pooling layers and a stroke average pooling layer. The encoder will extract stroke-level features from the input $\mathbf{X}$, which are represented as $\mathbf{S}$.

$$\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_{N_s}\}, \mathbf{s}_i \in \mathbb{R}^D \tag{4}$$

where $\mathbf{s}_i$ represents the feature of the i^{th} stroke.

The decoder consists of two GRU layers and a posterior attention mechanism. With $\mathbf{S}$ as input, the decoder outputs the probabilities of each category $P(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{S})$ at each step t. The output tokens are represented by $\mathbf{Y}$:

$$\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_C\}, \mathbf{y}_i \in \mathbb{R}^K \tag{5}$$

where $\mathbf{y}_i$ is a one-hot vector representing a symbol, K is the number of categories of symbols and C is the length of output string. Finally, $\mathbf{Y}$ is determined by the conditional distribution:

$$P(\mathbf{Y}|\mathbf{X}) = \prod_{t=1}^{C} P(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{S}) \tag{6}$$

B. Encoder With Stroke Average Pooling Layer

In our previous method TAP [2], features obtained by the encoder can be expressed as $\mathbf{P}$:

$$\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_{N_p/4}\}, \mathbf{p}_i \in \mathbb{R}^D \tag{7}$$

We call $\mathbf{P}$ as point-level features, because the number of features and the number of trajectory points are of the same order of magnitude. Because there are two pooling layers in the encoder, the number of point features is $N_p/4$. Point-level features have two disadvantages: First, each point-level feature contains insufficient contextual information. Also, there is a lot of redundant information between adjacent points. Although the features obtained through GRU have contextual information, the context information is insufficient when the sequence is too long. Second, the length of the output sequence is much less than that of point-level features. This is not only detrimental to the alignment between input and output, but more importantly, it is also not suitable for posterior probability attention [19].

To solve this problem, we propose a stroke average pooling layer to aggregate point-level features into stroke-level features. Since the stroke information can be obtained directly from the input device, this operation does not cause any additional cost of data labeling process. As shown in Fig. 1, we divide the trajectory points into strokes represented by different colors. The point-level features $\mathbf{P}$ are aggregated into the stroke-level features $\mathbf{S}$ through the stroke average pooling layer according to the stroke information $\mathbf{X}_{\text{mask}}$. The specific parallel calculation is expressed by the following formula:

$$\mathbf{X}_{\text{mask}} = \{\frac{\mathbf{m}_1}{||\mathbf{m}_1||_1}, \frac{\mathbf{m}_2}{||\mathbf{m}_2||_1}, \cdots, \frac{\mathbf{m}_{N_s}}{||\mathbf{m}_{N_s}||_1}\}, \mathbf{m}_i \in \mathbb{R}^{N_p/4} \tag{8}$$

$$\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_{N_s}\} = \mathbf{X}_{\text{mask}}^{\text{T}}\mathbf{P} \tag{9}$$

where $||\cdot||_1$ is L1-normalization and $\mathbf{m}_i$ is a vector of length $N_p/4$, which indicates which point-level features are included in the i^{th} stroke. If the j^{th} point-level feature belongs to the i^{th} stroke, the j^{th} element in $\mathbf{m}_i$ is set to 1, otherwise 0.

C. Decoder With Posterior Attention Mechanism

We denote the soft attention as $\mathbf{A}_t$ and posterior attention as $\hat{\mathbf{A}}_t$ at each time step t.

$$\mathbf{A}_t = [a_1^t, a_2^t, \cdots, a_{N_s}^t], a_i^t \in \mathbb{R} \tag{10}$$

$$\hat{\mathbf{A}}_t = [\hat{a}_1^t, \hat{a}_2^t, \cdots, \hat{a}_{N_s}^t], \hat{a}_i^t \in \mathbb{R} \tag{11}$$

We call $\hat{\mathbf{A}}_t$ as the posterior attention since $\hat{\mathbf{A}}_t$ is the attention distribution after observing the output label at the corresponding step. We expect posterior attention $\hat{\mathbf{A}}_t$ to be more accurate than soft attention $\mathbf{A}_t$ that is computed without knowledge of the output label at the corresponding step. Intuitively, also it makes sense because attention reflects an alignment of the input and the output, and its distribution will improve if the output is known.

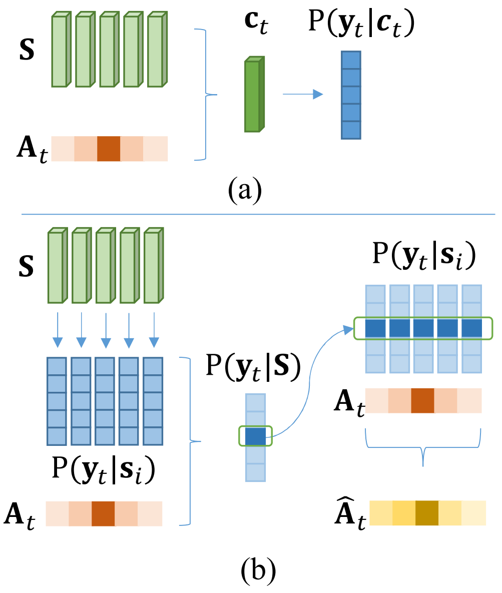
S
$\mathbf{c}_t$
$\text{P}(\mathbf{y}_t|\mathbf{c}_t)$
$\mathbf{A}_t$
(a)
S
$\text{P}(\mathbf{y}_t|\mathbf{S})$
$\text{P}(\mathbf{y}_t|\mathbf{s}_i)$
$\text{P}(\mathbf{y}_t|\mathbf{s}_i)$
$\mathbf{A}_t$
$\mathbf{A}_t$
$\hat{\mathbf{A}}_t$
(b)

Fig. 2. Simplified flowchart of soft attention mechanism (a) and posterior attention mechanism (b)

Compared with the soft attention mechanism, there are three changes in the posterior attention mechanism: First, $\mathbf{A}_t$ is calculated based on $\hat{\mathbf{A}}_{t-1}$.

$$\mathbf{A}_t = \text{F}_{\text{att}}(\mathbf{h}_{t-1}, [\mathbf{y}_{t-1}, \hat{\mathbf{c}}_{t-1}], \sum_{l=1}^{t-1} \hat{\mathbf{A}}_l, \mathbf{S}) \tag{12}$$

$$\hat{\mathbf{c}}_{t-1} = \sum_{i=1}^{Ns} \hat{a}_i^{t-1}\mathbf{s}_i \tag{13}$$

where F_{att} is the function [2] that calculates the soft attention distribution and $\mathbf{h}_{t-1}$ denotes the previous hidden state of decoder. $\hat{\mathbf{c}}_{t-1}$ is the context vector at time step $t-1$ calculated based on the posterior attention $\hat{\mathbf{A}}_{t-1}$ and is concatenated with the previous target token $\mathbf{y}_{t-1}$ to get a new vector. All the elements of $\hat{\mathbf{A}}_0$ are initialized to 0. Then the previous hidden state $\mathbf{h}_{t-1}$ together with the concatenated vector $[\mathbf{y}_{t-1}, \hat{\mathbf{c}}_{t-1}]$ are used to compute the query of the attention, and the stroke-level features $\mathbf{S}$ denotes the key of the attention. Besides, we append a coverage vector for attention mechanism, which is computed with the summation of all past posterior attention probabilities $\sum_{l=1}^{t-1} \hat{\mathbf{A}}_l$, to address the over-parsing or under-parsing problems. Second, the place where attention performs on is changed from the input to the output as shown in Fig. 2. More specifically, $\mathbf{A}_t$ was used as a weight to sum $\text{P}(\mathbf{y}_t|\mathbf{h}_t, \mathbf{s}_i)$

to get the output probability distribution $\mathrm{P}(\mathbf{y}_t|\mathbf{y}_{t-1},\mathbf{S})$, instead of summing $\mathbf{s}_i$ to get the context vector $\mathbf{c}_t$:

$$\mathrm{P}(\mathbf{y}_t|\mathbf{y}_{t-1},\mathbf{S}) = \sum_{i=1}^{N_s} a_i^t \mathrm{P}(\mathbf{y}_t|\mathbf{h}_t,\mathbf{y}_{t-1},\mathbf{s}_i) \tag{14}$$

Third, the posterior attention distribution $\hat{\mathbf{A}}_t$ propagated to the next decoding step is conditioned on the output.

$$\hat{a}_i^t = \frac{a_i^t \mathrm{P}(y_t|\mathbf{h}_t,\mathbf{s}_i)}{\sum_k a_k^t \mathrm{P}(y_t|\mathbf{h}_t,\mathbf{s}_k)} \tag{15}$$

The other calculation process in the decoder is the same as TAP [2].

III. Experiments

In order to verify the effectiveness of stroke based posterior attention for OHMER, we conduct several sets of experiments on the CROHME competition dataset [4], [10].

A. Datasets

We validated the proposed model on CROHME 2014 test set [4] and CROHME 2016 test set [10]. The CROHME competition dataset is currently the most widely used public dataset for online handwritten mathematical expression recognition. The training set has 8,836 expressions including 101 math symbol classes. The CROHME 2014 test set has 986 expressions and the CROHME 2016 test set has 1,147 expressions.

B. Details of Training and inference

1) Model Parameters: To be fairly comparable, the parameters of the proposed model are the same as TAP [2]. Encoder consists of 4 bidirectional GRU layers and each GRU layer contains 256 GRU units. The third and fourth layers are followed by a max pooling operation with kernel of size 2×1. In decoder, the embedding dimension and the GRU decoder dimension are set to 256. The attention dimension is set to 500, and the convolution used for processing the history of attention has a kernel of 7×1 with dimension 256. The dropout layers [22] are adopted to alleviate the problem of overfitting with drop ratio set to 0.2.

2) Loss Functions: Our goal is to maximize the conditional distribution $P(\mathbf{Y}|\mathbf{X})$. Cross entropy criterion is selected as the objective function to calculate the cost:

$$O_p = -\sum_{t=1}^{C} \log \mathrm{P}(y_t|\mathbf{y}_{t-1},\mathbf{X}) \tag{16}$$

where y_t is the ground truth token at time step t. When we use attention guider [2] as a supervision of the posterior attention mechanism, another loss function O_a needs to be added:

$$O_a = \sum_{t=1}^{C} G_t \tag{17}$$

$$O = O_p + \lambda O_a \tag{18}$$

where G_t is the cost of attention guider at time step t and λ is set to 0.2.

TABLE I
Comparison of recognition performance (in %) on CROHME 2014 and CROHME 2016 between system I to IV

System	Attention	Feature	CROHME 2014		CROHME 2016	
		Level	WER	ExpRate	WER	ExpRate
I	soft	point	13.34	50.71	14.67	45.95
II	posterior	point	11.97	51.28	13.21	47.28
III	soft	stroke	13.29	50.91	14.53	47.60
IV	posterior	stroke	10.44	54.26	12.68	51.75

3) Optimization and Inference Strategy: We adopted the ADADELTA algorithm [23] as the optimizer and used the weight noise [24] as the regularization. We set the weight decay to 10-5, the learning rate to 1, the parameter ρ and ϵ of the optimizer to 0.95 and $1e^{-8}$. During the training process, we adopted an early-stopping training strategy. Whenever the word error rate (WER) [25] of the validation set does not decrease for 15 consecutive epochs, the learning rate will decay to one tenth of the current. When the learning rate decays three times, the model will stop training, and the weight of the model with the lowest WER will be saved as the final result. Beam search algorithm is employed to implement the decoding procedure. Here, we maintained a set of 10 partial hypotheses at each time step.

4) Metric: We use expression recognition rates (ExpRate) and WER to evaluate our system. ExpRate is the percentage of predicted mathematical expressions matching the ground truth which is a stricter evaluation standard than WER. When comparing with other systems, we also calculated ExpRates with at most one to three symbol-level errors using official tools [10].

C. Evaluation of Posterior Attention Mechanism

As shown in Table I, our reproduced system I (TAP) achieves an ExpRate of 50.71% on CROHME 2014 test set and an ExpRate of 45.95% on CROHME 2016 test set. System II uses the posterior attention mechanism and only improves the ExpRate by only 0.57% on CROHME 2014 test set and 1.33% on CROHME 2016 test compared with system I. This proves that merely adding the posterior attention mechanism does not bring a significant improvement. The reason is that point-level features are not enough to describe math symbols, thus the posterior information is not sufficient. Obviously the point-level features are not suitable for the posterior attention mechanism.

Then, we add the stroke average pooling layer to the encoder in system I to build system III, which achieves an ExpRate of 50.91% on CROHME 2014 test set and an ExpRate of 47.60% on CROHME 2016 test set. Compared with system I, system III based on stroke-level features achieves slightly better performance. System IV uses the stroke based posterior attention mechanism and achieves an ExpRate of 54.26% on CROHME 2014 test set and an ExpRate of 51.75% on CROHME 2016 test set. Compared with the previous three systems, system IV achieves the best results and improves the

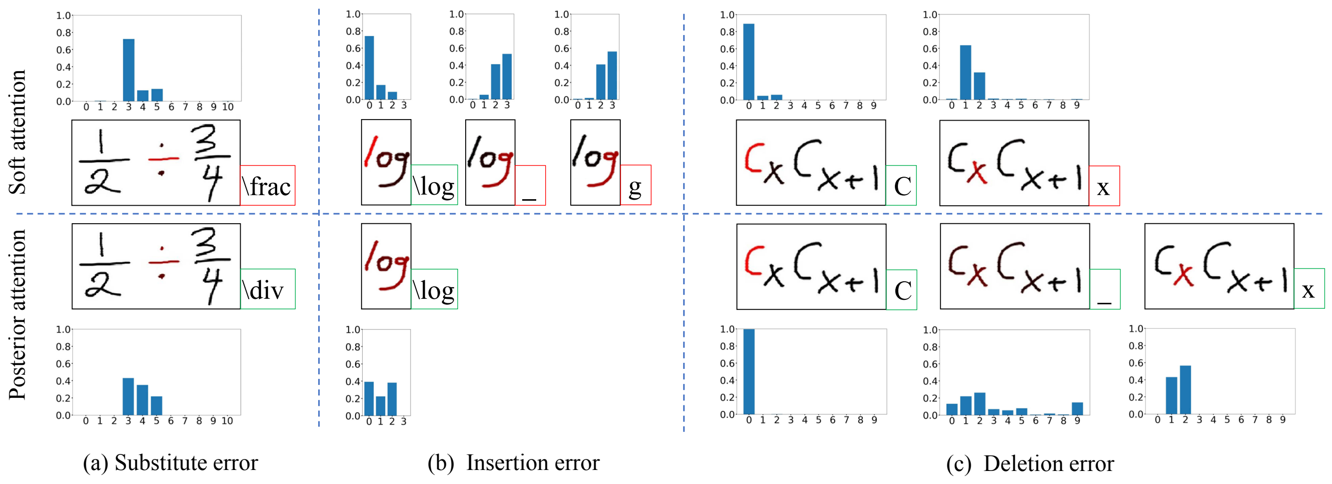

Fig. 3. Three examples of attention visualization with soft attention and posterior attention. The horizontal axis of the histogram represents the serial number of the strokes, and the vertical axis represents the value of the attention weight.

ExpRate by 3.55% on CROHME 2014 test set and 5.80% on CROHME 2016 test. It is evident that the stroke-level features are more suitable for the posterior attention mechanism and the proposed stroke based posterior attention can bring significant improvements.

D. Attention Visualization

In order to better explain the superiority of the posterior attention mechanism, we show three examples of attention visualization with soft attention and posterior attention. As shown in Fig. 3, we connect the track points and display them on a white background. We use red to represent the attention probabilities, where the darker color describes the higher attention probabilities. In order to show the difference between the two distributions of attention weights more intuitively, we draw a histogram of each distribution. The horizontal axis of the histogram represents the serial number of the strokes, and the vertical axis represents the value of the attention weight. As shown in Fig. 3 (a), the division sign "÷" is incorrectly identified as a fraction sign "−" because the soft attention weights are concentrated on the horizontal line of the division sign. The posterior attention weights are evenly distributed over the entire division sign, so the division sign is correctly recognized. In Fig. 3 (b), the log symbol "log" is parsed three times by the soft attention mechanism and recognized as "$\log_g$", which caused an insertion error. In stroke based posterior attention, the log symbol is only parsed once by the posterior attention mechanism and correctly recognized. In Fig. 3 (c), the soft attention mechanism ignores the spatial relationship between "C" and "x", so the subscript sign "_" is not obtained during the decoding process. The posterior attention mechanism pays attention to both "C" and "x", and recognizes the correct spatial relationship. These three examples prove that the posterior attention mechanism can learn more accurate and reasonable alignment information.

TABLE II
COMPARISON OF RECOGNITION PERFORMANCE (IN %) ON CROHME 2014 BETWEEN SYSTEM I TO IV WITHOUT ATTENTION GUIDER

System	Attention	Feature Level	ExpRate Without Guider	ExpRate With Guider
I	soft	point	48.24	50.71
II	posterior	point	50.24	51.28
III	soft	stroke	48.28	50.91
IV	posterior	stroke	53.85	54.26

E. Ablation Experiments of Attention Guider

Because the attention guider needs the ground truth of the alignment information during the training process, which leads to high cost of data labeling process. We hope that the posterior attention mechanism can still perform well without attention guider. As shown in Table II, we used the results of system I in Section III-C as a baseline and run 4 experiments without attention guider to compare. System I without attention guider only achieves an ExpRate of 48.24% on CROHME 2014 test set, which is 2.47% lower than the baseline. Same as system I, ExpRates of both system II and system III decrease and are lower than the baseline when the attention guider is not used. However, the system IV without attention guider achieves an ExpRate of 53.85% on CROHME 2014 test set, which is only 0.41% lower than the system IV in Section III-C. It is encouraging that the posterior attention mechanism can still achieve excellent results without using attention guider. These experiments prove that the proposed stroke based posterior attention has better robustness and generality.

F. Comparison with State-of-the-arts

In order to prove that the stroke based posterior attention can achieve state-of-the-art results, we compared our model with other models on several test sets of CROHME [4], [10]. For fair comparison, our model does not use additional training sets and additional formula corpus. All experimental

TABLE III
COMPARISON OF EXPRATE (IN %) ON CROHME 2014 AND CROHME 2016

System	CROHME 2014				CROHME 2016			
	ExpRate	≤ 1	≤ 2	≤ 3	ExpRate	≤ 1	≤ 2	≤ 3
Wiris [10]	-	-	-	-	49.61	60.42	64.69	-
Tokyo [10]	-	-	-	-	43.94	50.91	53.70	-
Merge 9 [26]	29.91	39.94	44.96	50.15	27.03	35.48	42.46	-
PGS [27]	48.78	66.13	**73.94**	**79.01**	45.60	62.25	**70.44**	**75.76**
TAP	50.71	65.42	68.73	69.54	45.95	60.77	63.85	64.57
Res-BiRNN [28]	53.35	64.50	70.08	72.92	47.95	60.16	65.56	68.61
Ours	**54.26**	**69.64**	72.65	73.26	**51.75**	**65.18**	68.27	68.99

results of our model are obtained from a single model and higher ExpRate can be achieved through an ensemble method [29]. When comparing with other systems, we also calculated ExpRates with at most one to three symbol-level errors using official tools [10]. As shown in Table III, we chose two systems that participated in CROHME 2016 competition [10] and three recently published systems. The system Wiris is the best system on CROHME 2016 competition using only official training dataset. Note that it used a Wikipedia formula corpus, consisting of more than 592,000 mathematical expressions, to train a strong language model. The system Merge 9 [26] is a tree-BLSTM-based recognition system. The system PGS [27] uses pattern generation strategies to augment the training data. The system Res-BiRNN [28] employ residual connection in the BiRNN layers to improve feature extraction. Since the results shown in [2] are the ensemble of three TAP models, we show the results of System I in Section III-C. We can see that our proposed system still achieves the best result with ExpRate of 54.26% on CROHME 2014 and ExpRate of 51.75% on CROHME 2016 without using additional corpus and data augmentation strategies. It is evident that the proposed stroke based posterior attention exhibits higher performance than previous methods.

IV. CONCLUSION

In this study we introduce an end-to-end framework with stroke average pooling layer and posterior attention mechanism to recognize online handwritten mathematical expressions. Through experimental analysis and attention visualization, we demonstrate that the posterior attention mechanism is better than soft attention mechanism and only stroke-level posterior attention can perform well, proving the hypothesis that only the feature vectors which contain enough classification information can calculate posterior attention accurately. In the future, we will continue to explore how to apply the posterior attention mechanism in other handwritten recognition problems and how to improve the attention mechanism.

ACKNOWLEDGMENT

This work was supported in part by the National Key R&D Program of China under contract No. 2017YFB1002202, the National Natural Science Foundation of China under Grants No. 61671422 and U1613211, the Key Science and Technology Project of Anhui Province under Grant No. 17030901005. This work was also funded by MOE-Microsoft Key Laboratory of USTC.

REFERENCES

[1] 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.

[2] J. Zhang, J. Du, and L. Dai, "Track, attend, and parse (tap): An end-to-end framework for online handwritten mathematical expression recognition," *IEEE Transactions on Multimedia*, vol. 21, no. 1, pp. 221–233, 2018.

[3] H. Büyükbayrak, B. Yanikoglu, and A. Erçil, "Online handwritten mathematical expression recognition," in *Document Recognition and Retrieval XIV*, vol. 6500. International Society for Optics and Photonics, 2007, p. 65000F.

[4] H. Mouchere, R. Zanibbi, U. Garain, and C. Viard-Gaudin, "Advancing the state of the art for handwritten math recognition: the crohme competitions, 2011–2014," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 19, no. 2, pp. 173–189, 2016.

[5] F. Yin, Q.-F. Wang, X.-Y. Zhang, and C.-L. Liu, "Icdar 2013 chinese handwriting recognition competition," in *2013 12th International Conference on Document Analysis and Recognition*. IEEE, 2013, pp. 1464–1470.

[6] U. Garain and B. B. Chaudhuri, "Recognition of online handwritten mathematical expressions," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 6, pp. 2366–2376, 2004.

[7] F. Simistira, V. Katsouros, and G. Carayannis, "Recognition of online handwritten mathematical formulas using probabilistic svms and stochastic context free grammars," *Pattern Recognition Letters*, vol. 53, pp. 85–92, 2015.

[8] J. Zhang, J. Du, and L. Dai, "A gru-based encoder-decoder approach with attention for online handwritten mathematical expression recognition," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 902–907.

[9] J. Wang, J. Du, and J. Zhang, "Stroke constrained attention network for online handwritten mathematical expression recognition," *arXiv preprint arXiv:2002.08670*, 2020.

[10] H. Mouchère, C. Viard-Gaudin, R. Zanibbi, and U. Garain, "Icfhr2016 crohme: Competition on recognition of online handwritten mathematical expressions," in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2016, pp. 607–612.

[11] 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.

[12] B. Huang and M. Kechadi, "A structural analysis approach for online handwritten mathematical expressions," *Int. J. Comput. Sci. Netw. Secur*, vol. 7, pp. 47–56, 2007.

[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*, vol. 71, pp. 196–206, 2017.

[14] A. D. Le and M. Nakagawa, "Training an end-to-end system for handwritten mathematical expression recognition by generated patterns," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 1056–1061.

[15] J. Wang, J. Du, J. Zhang, and Z.-R. Wang, "Multi-modal attention network for handwritten mathematical expression recognition," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019, pp. 1181–1186.

[16] J.-W. Wu, F. Yin, Y.-M. Zhang, X.-Y. Zhang, and C.-L. Liu, "Handwritten mathematical expression recognition via paired adversarial learning," *International Journal of Computer Vision*, pp. 1–16, 2020.

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

[18] 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.

[19] S. Shankar and S. Sarawagi, "Posterior attention models for sequence to sequence learning," in *International Conference on Learning Representations*, 2018.

[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] X.-Y. Zhang, F. Yin, Y.-M. Zhang, C.-L. Liu, and Y. Bengio, "Drawing and recognizing chinese characters with recurrent neural network," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 849–862, 2017.

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

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

[24] A. Graves, "Practical variational inference for neural networks," in *Advances in neural information processing systems*, 2011, pp. 2348–2356.

[25] D. Klakow and J. Peters, "Testing the correlation of word error rate and perplexity," *Speech Communication*, vol. 38, no. 1-2, pp. 19–28, 2002.

[26] T. Zhang, H. Mouchère, and C. Viard-Gaudin, "A tree-blstm-based recognition system for online handwritten mathematical expressions," *Neural Computing and Applications*, pp. 1–20, 2018.

[27] A. D. Le, B. Indurkhya, and M. Nakagawa, "Pattern generation strategies for improving recognition of handwritten mathematical expressions," *Pattern Recognition Letters*, vol. 128, pp. 255–262, 2019.

[28] Z. Hong, N. You, J. Tan, and N. Bi, "Residual birnn based seq2seq model with transition probability matrix for online handwritten mathematical expression recognition," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019, pp. 635–640.

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