Track, Attend and Parse (TAP): An End-to-end Framework for Online Handwritten Mathematical Expression Recognition

Jianshu Zhang, Jun Du, and Lirong Dai

***Abstract*—In this paper, we introduce Track, Attend and Parse (TAP), an end-to-end approach based on neural networks for online handwritten mathematical expression recognition (OHMER). The architecture of TAP consists of a tracker and a parser. The tracker employs a stack of bidirectional recurrent neural networks with gated recurrent units (GRU) to model the input handwritten traces, which can fully utilize the dynamic trajectory information in OHMER. Followed by the tracker, the parser adopts a GRU equipped with guided hybrid attention (GHA) to generate LaTeX notations. The proposed GHA is composed of a coverage based spatial attention, a temporal attention and an attention guider. Moreover, we demonstrate the strong complementarity between offline information with static-image input and online information with ink-trajectory input by blending a fully convolutional networks based watcher into TAP. Inherently unlike traditional methods, this end-to-end framework does not require the explicit symbol segmentation and a predefined expression grammar for parsing. Validated on a benchmark published by the CROHME competition, the proposed approach outperforms the state-of-the-art methods and achieves the best reported results with an expression recognition accuracy of 61.16% on CROHME 2014 and 57.02% on CROHME 2016, using only official training dataset.**

***Index Terms*—Online handwritten mathematical expression recognition, end-to-end framework, gated recurrent unit, guided hybrid attention, ensemble.**

I. Introduction

MATHEMATICAL expressions play an important role in scientific documents. They are indispensable for describing problems and theories in maths, physics and many other fields. Meanwhile, people have begun to use handwritten mathematical expressions (HMEs) as one natural input mode. However, how to successfully recognize them remains a difficult problem as online handwritten mathematical expression recognition (OHMER) exhibits three distinct challenges [1], [2], i.e., the complicated two-dimensional structures, enormous ambiguities in handwriting input and strong dependency on contextual information.

Technically, OHMER consists of two major problems [3], namely symbol recognition and structural analysis, which can be solved sequentially or globally. Sequential solutions [4], [5] first segment input expression into math symbols and recognize them separately. The analysis of two-dimensional structures is then carried out based on the best symbol segmentation and symbol recognition results. In sequential solutions, symbol recognition does not make use of contextual information which could be helpful to reduce the ambiguities of handwritten symbols. Besides, the symbol segmentation and recognition errors will be subsequently inherited by structural analysis. Conversely, global solutions [6], [7] seem to be more appropriate as symbol recognition and structural analysis are optimized using the global information of expression simultaneously. However, global solutions are computationally more expensive because the probabilities for segmentation composed of strokes (a sequence of points between a pen-down and a pen-up operation) are exponentially expanded. Therefore effective search strategies must be executed [8]. Specific to structural analysis, many approaches [9], [10], [11], [12] have been investigated. Among them, the grammar-based approaches are widely used [13], [14]. Structural analysis with a grammar is often termed as syntactic pattern recognition but these grammars are constructed using extensive prior knowledge. Also, [15], [16] proposed more general and simpler methods for structural analysis without using a math grammar, termed as structural pattern recognition. They use spanning trees to select parse tree directly from hypothesis graphs.

Inspired by recent work in neural networks [17], [18], [19], we introduce a framework totally based on neural networks for OHMER. The proposed approach is named as Track, Attend and Parse (TAP), consisting of a tracker and a parser equipped with guided hybrid attention (GHA). The proposed TAP possesses three distinctive properties: 1) It is end-to-end trainable; 2) It is data-driven, which means it does not require a predefined math grammar; 3) Symbol segmentation can be automatically performed through attention mechanism. Unlike conventional methods, which recognize HMEs as expression trees, TAP learns to track the traces of HMEs and parse them as LaTeX [20] notations. We employ a stack of bidirectional recurrent neural networks with gated recurrent units (GRU) [21] as the tracker which takes the two-dimensional handwritten traces as input and maps the trajectory information to high-level representations. The parser is implemented by a unidirectional GRU with GHA which converts these high-level representations into output LaTeX strings, one symbol at a time. Here, GHA is composed of a coverage based spatial attention, a temporal attention and an attention guider. More specifically, for each predicted symbol, the spatial attention scans the entire input (traces of HME) and teaches the parser where to attend for describing a math symbol or an implicit spatial operator.

Jianshu Zhang, Jun Du and Lirong Dai were with the National Engineering Laboratory for Speech and Language Information Processing, University of Science and Technology of China, Hefei, Anhui, P. R. China. e-mail: xysszjs@mail.ustc.edu.cn, jundu@ustc.edu.cn, lrdai@ustc.edu.cn. (Corresponding author: Jun Du.)

Meanwhile, the temporal attention tells the parser when to rely on the product of spatial attention and when to just rely on the language model built in the parser. Because in LaTeX notations, some symbols can often be predicted reliably just from the language model, e.g., the symbol "{" in string "x ∧ { 2 } + 1" (LaTeX notation of $x^2 + 1$). During the training procedure, the learning of spatial attention can also be guided by an attention guider performing as a regularization term. Inherently different from traditional approaches, our model optimizes symbol segmentation automatically through the attention mechanism, and structural analysis does not rely on a predefined grammar as a data-driven language model is built in the parser. Based on this end-to-end framework, the two typical problems, namely symbol recognition and structural analysis, are jointly optimized. Moreover, we investigate on two modalities to express the HMEs: the static images (offline information) or the dynamic traces (online information). The strong complementarity between offline and online information is demonstrated by blending TAP with a fully convolutional networks (FCN) [22], [23] based watcher to handle the static HME images. Finally, we incorporate a stronger GRU based language model trained on an additional text dataset provided by CROHME competition to further improve the recognition performance.

The main contributions of this study can be summarized as:

- A novel TAP framework is proposed for OHMER, which is an end-to-end trainable model to alleviate the problems caused by symbol segmentation and computational demands of employing a math grammar in the conventional approaches.
- A hybrid attention with an attention guider is incorporated with TAP to show where and when to attend.
- TAP is blended with a FCN-based watcher and a stronger GRU based language model to fully utilize the offline information of OHME and the contextual information.
- We experimentally demonstrate how TAP completes the automatic symbol segmentation and structural analysis through visualization of hybrid attention.

II. Related works

In this section, we describe the previous work on OHMER, including both traditional grammar based approaches and recent neural network based approaches.

A. Grammar based approaches for OHMER

Symbol recognition and structural analysis are two basic components for OHMER. In the data acquisition of online handwriting, the pen-tip movements (xy-coordinates) and pen states (pen-down or pen-up) are automatically stored as variable-length sequential data. Meanwhile, the sequential data can also be transformed into image-like representations as shown in Fig. 1. Inspired by recent work in neural networks, convolutional neural networks (CNN) [24] and recurrent neural networks (RNN) [25] have been widely used as powerful classifiers for offline or online symbol recognition. However, regarding to structural analysis, many researchers prefer approaches based on predefined grammars

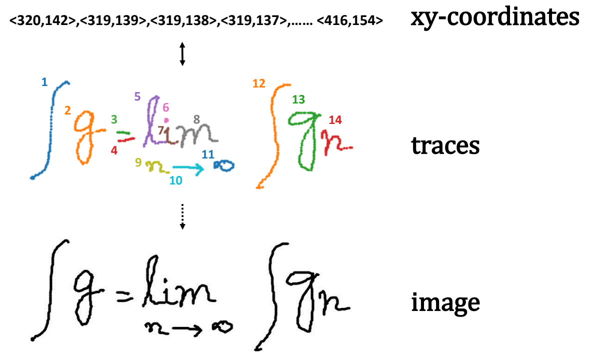


Fig. 1. An example of online handwritten mathematical expression, including the sequential data, visualization of traces and its transformed 2D static image.

as a natural way to solve the problem. Different types of math grammars have been investigated. For example, Chan and Yeung [26] used definite clause grammars. Álvaro et al [5], [7] showed the effectiveness of stochastic context-free grammars on several systems as they consistently performed best in the CROHME competitions. Yamamoto et al. [27] also presented an OHMER system using probabilistic context-free grammars, and MacLean [28] developed an approach using relational grammars and fuzzy sets.

B. Encoder-decoder framework

This study pays a special attention to neural network based structural analysis. In [18], a novel neural network framework, namely encoder-decoder, was exploited to address sequence to sequence learning. Typically, both the encoder and the decoder are implemented with RNNs, which are proved to be good processor and generator for sequential signals. The encoder RNN first learns to encode the sequential variable-length input into high-level representations. A fixed-length context vector is then generated via summing the variable-length representations or just choosing the last representation. Finally, the decoder RNN uses this context vector to generate variable-length output sequence, one word at a time. Due to the intermediate fixed-length vector, the encoder-decoder model can well perform a mapping between variable-length input and output. This framework has been extensively applied to many applications including machine translation [29], [30], speech recognition [31], [32], character recognition [33], [34], image captioning [35], [36] and video processing [37], [38].

C. Attention

In [39], [40], [41], attention was shown to be one of the most distinct aspects in human visual system. This mechanism could be incorporated into encoder-decoder framework for calculating the fixed-length context vector, i.e., the variable-length representations could be summed using the attention as the weighting coefficients. After adopting an attention mechanism into encoder-decoder, salient regions in the static representation can dynamically rise to the forefront. The attention mechanism plays an indispensable role in image captioning for obtaining a state-of-the-art performance. For example, [42] proposed the "hard" attention for image captioning system to know where to attend and its effectiveness was shown through attention visualization. [43] proposed the concept of "attention correctness" to strength the alignment for image captioning. In [44], an adaptive attention was implemented via a visual sentinel so that the captioning system could also know when to attend, and with a similar motivation, [45] also proposed a global-local attention so that model could selectively pay attention to spatial objects and context information.

D. Neural network based approaches for OHMER

The generality of the attention based encoder-decoder framework suggests that OHMER may also be one proper application. Recently, [46], [47] used the attention based encoder-decoder model for OHMER and significantly outperformed the best system on CROHME 2014. In [46] the proposed model consisted of a FCN encoder and a GRU decoder equipped with a coverage-based attention model while [47] employed a CRNN as the encoder and the decoder is equipped with a coarse-to-fine attention model. However, both [46] and [47] treated the HMEs input as static images which ignores the handwriting dynamics (namely the temporal order and trajectory). As we can see in Fig. 1, besides the symbol shape information, the writing order is also preserved in the online sequential data, which is important information and can not be recovered from the static image. Therefore, to capture the dynamic information to reduce handwritten ambiguities, [48] proposed to employ a GRU encoder that directly takes the raw sequential data as input. Validated on CROHME 2014, [48] showed a significant improvement of recognition accuracy over [46], [47].

This study is an extension of the previous work in [48] with the following new contributions. 1) We propose to employ a temporal attention to teach the parser when to rely on the representations extracted by tracker and when to just rely on the built-in language model. 2) To compute the temporal attention, the spatial attention is slightly adjusted. Meanwhile, we newly introduce an attention guider to help improve the learning of spatial attention. 3) We blend a FCN watcher into TAP by considering the strong complementarity between static-image based input and dynamic-trace based input. By processing HMEs from two different modalities, the strengths of [46] and [48] can be fully utilized simultaneously. 4) We use an extra official text dataset containing only LaTeX notations to train an additional language model for enhancing our parser. 5) More experiments and analyses are included.

III. Network Architecture of TAP

In this section, we elaborate the proposed TAP architecture which parses a mathematical expression structure into a LaTeX string by tracking a sequence of online handwritten points. As illustrated in Fig. 2, the raw data is a sequence of points containing xy-coordinates which can be visualized as the bottom-right image by drawing the trajectory. A preprocessing is first applied to extract trajectory information from raw sequential data. The tracker is a stack of bidirectional GRU while the parser combines a GRU based language model and a hybrid attention mechanism. As for the hybrid attention

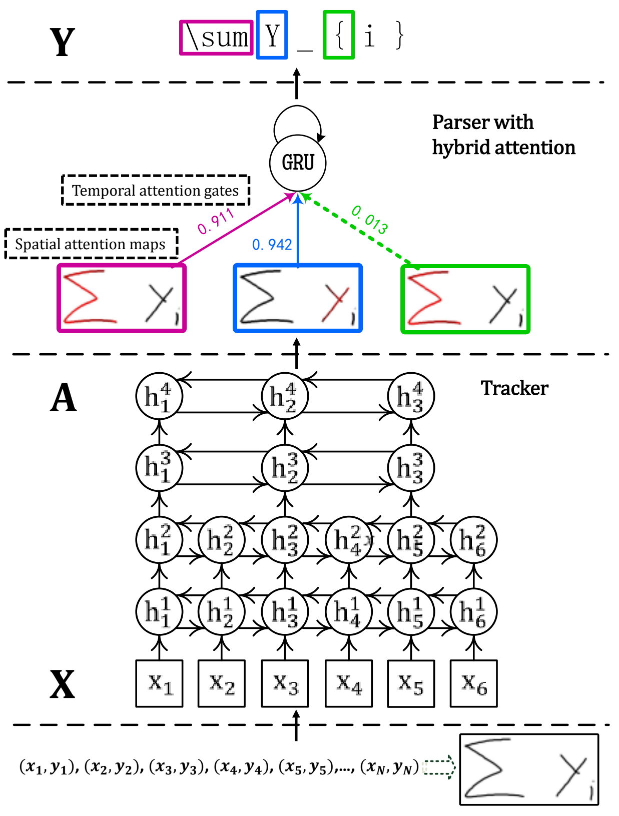

Y
\sum
Y
_
{
i
}
Parser with hybrid attention
GRU
Temporal attention gates
Spatial attention maps
0.911
0.942
0.013
A
Tracker
h_1^4 h_2^4 h_3^4
h_1^3 h_2^3 h_3^3
h_1^2 h_2^2 h_3^2 h_4^2 h_5^2 h_6^2
h_1^1 h_2^1 h_3^1 h_4^1 h_5^1 h_6^1
X
x_1 x_2 x_3 x_4 x_5 x_6
$(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5), \dots, (x_N, y_N)$

Fig. 2. Overall architecture of Track, Attend and Parse. $\mathbf{X}$ denotes the input sequence in Section III-A, $\mathbf{A}$ denotes the annotation sequence in Section III-C, $\mathbf{Y}$ denotes the output sequence in Section III-C.

mechanism, spatial attention can potentially well learn the alignment between input traces and output string while temporal attention can well know when to rely on the product of spatial attention and when to just rely on the language model. For example, in Fig. 2, the purple, blue and green rectangles denote three symbols with the red color representing the spatial attention probabilities of each handwritten symbol (lighter color denotes higher probability) and the probabilities linking to rectangles represent their reliability produced by temporal attention. When predicting the math symbol "\sum", the spatial attention model aligns well to the stroke of "$\sum$" (in the purple spatial attention map) which corresponds to the human intuition and the temporal attention probability linking to the purple rectangle is extremely high as the spatial attention map is accurate. Conversely, when predicting the math symbol "{", there is no object for spatial attention model to attend to, leading to an inaccurate spatial attention map. Therefore the temporal attention probability linking to the green spatial attention map is small which tells the parser should rely on the built-in language model at this time.

A. Preprocessing

Suppose the input traces contain raw sequential data with a variable-length N:

$$\{[x_1, y_1, s_1], [x_2, y_2, s_2], \dots, [x_N, y_N, s_N]\} \qquad (1)$$

where x_i and y_i are xy-coordinates of the pen-tip movements, s_i is the stroke index of the i^{th} point, and the sequence is stored in the writing process.

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 sequential points are first operated according to [49]. Then we extract an 8-dimensional feature vector for each point:

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

where $\Delta x_i = x_{i+1} - x_i$, $\Delta y_i = y_{i+1} - y_i$, $\Delta' x_i = x_{i+2} - x_i$, $\Delta' y_i = y_{i+2} - y_i$ and $\delta(\cdot) = 1$ when the condition is true or otherwise zero. The last two terms are flags indicating the status of the pen, i.e., [1, 0] means pen-down while [0, 1] means pen-up. Note that, an HME can also be considered as a sequence of several strokes. So fully utilizing the stroke status information plays an important role in increasing recognition accuracy. 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 tracker, but note that each item $\mathbf{x}_i$ here is actually the 8-dimensional vector shown in Eq. (2).

B. Tracker

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

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

where $\mathbf{W}_{xh}$ is the connection weight matrix of the network between input layer and hidden layer, and $\mathbf{U}_{hh}$ is the weight matrix of recurrent connections in the same hidden layer. In principle, the recurrent connections let 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 [50], [51].

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

$$\mathbf{h}_t = \text{GRU}\left(\mathbf{x}_t, \mathbf{h}_{t-1}\right) \tag{4}$$

as illustrated in Fig. 3 the GRU function can be expanded as follows:

$$\mathbf{z}_t = \sigma(\mathbf{W}_{xz}\mathbf{x}_t + \mathbf{U}_{hz}\mathbf{h}_{t-1}) \tag{5}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_{xr}\mathbf{x}_t + \mathbf{U}_{hr}\mathbf{h}_{t-1}) \tag{6}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{U}_{rh}(\mathbf{r}_t \otimes \mathbf{h}_{t-1})) \tag{7}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \otimes \mathbf{h}_{t-1} + \mathbf{z}_t \otimes \tilde{\mathbf{h}}_t \tag{8}$$

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

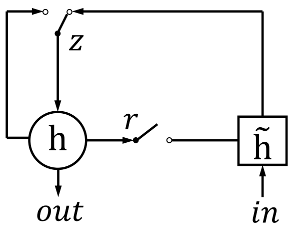


Fig. 3. Illustration of GRU function, $\mathbf{z}$ denotes update gate, $\mathbf{r}$ denotes reset gate, $\tilde{\mathbf{h}}$ denotes candidate activation and $\mathbf{h}$ denotes the output activation.

Nevertheless, unidirectional GRU cannot utilize the future context. Accordingly, 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 high-level representation, the tracker stacks multiple layers of GRUs on top of each other. However, with the increased depth, the high-level representation is overly precise for the parser and contains much redundant information (difference of feature vectors between adjacent points is slight). This leads us to add pooling over points in high-level GRU layers as shown in Fig. 2. The pooling is a subsampling operation. We drop the even outputs of lower layer and only send the odd outputs to upper layer, therefore the upper layers run twice faster than the lower ones. Besides accelerating the tracking process, the pooling operation also improves the recognition performance as it is easier for the parser to implement spatial attention with a fewer number of outputs of tracker.

C. Parser

In Fig. 2, the parser generates a corresponding LaTeX notation of the input traces. The output string $\mathbf{Y}$ is represented by a sequence of one-hot encoded symbols.

$$\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_C\} \ , \ \mathbf{y}_i \in \mathbb{R}^K \tag{9}$$

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

Meanwhile, assuming that the tracker extracts high-level representations denoted by an annotation sequence $\mathbf{A}$ with length L. If there is no pooling in the stacked GRU, $L = N$ (N is the length of input sequential data); otherwise N will be several multiples of L and each of these annotations represents a D-dimensional vector corresponding to a local region of original traces:

$$\mathbf{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_L\} \ , \ \mathbf{a}_i \in \mathbb{R}^D \tag{10}$$

Note that, both the length of annotation sequence L and the length of LaTeX string 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 $\mathbf{c}_t$ by employing guided hybrid attention which will be described in more details in Section III-D. Given the context vector $\mathbf{c}_t$, we utilize unidirectional GRU to produce LaTeX strings symbol by symbol. The probability of each predicted symbol is computed

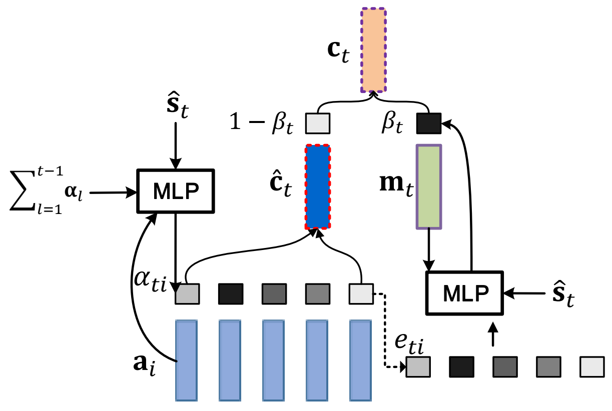


Fig. 4. Schematic representation of hybrid attention, consisting of a coverage based spatial attention and a temporal attention.

by the context vector $\mathbf{c}_t$, current GRU hidden state $\mathbf{s}_t$ and previous target symbol $\mathbf{y}_{t-1}$ using the following equation:

$$p(\mathbf{y}_t|\mathbf{y}_{t-1},\mathbf{X}) = g\left(\mathbf{W}_o h(\mathbf{E}\mathbf{y}_{t-1}+\mathbf{W}_s\mathbf{s}_t+\mathbf{W}_c\mathbf{c}_t)\right) \quad (11)$$

where g denotes a softmax activation function over all the symbols in the vocabulary, h denotes a maxout activation function, $\mathbf{W}_o \in \mathbb{R}^{K\times\frac{m}{2}}$, $\mathbf{W}_s \in \mathbb{R}^{m\times n}$, $\mathbf{W}_c \in \mathbb{R}^{m\times D}$, and $\mathbf{E}$ denotes the embedding matrix, m and n are the dimensions of embedding and GRU parser.

The parser adopts two unidirectional GRU layers to calculate the hidden state $\mathbf{s}_t$:

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

$$\mathbf{c}_t = f_{\mathrm{hatt}}\left(\mathbf{y}_{t-1},\mathbf{s}_{t-1},\hat{\mathbf{s}}_t,\mathbf{A}\right) \quad (13)$$

$$\mathbf{s}_t = \mathrm{GRU}\left(\mathbf{c}_t,\hat{\mathbf{s}}_t\right) \quad (14)$$

where $\mathbf{s}_{t-1}$ denotes the previous hidden state, f_{hatt} denotes the hybrid attention model, and $\hat{\mathbf{s}}_t$ is the prediction of current GRU hidden state. The initial hidden state $\mathbf{s}_0$ of the first GRU is predicted by an average of annotation vectors $\mathbf{a}_i$ fed through a fully-connection layer :

$$\bar{\mathbf{a}} = \frac{1}{L}\sum_{i=1}^{L}\mathbf{a}_i \quad (15)$$

$$\mathbf{s}_0 = \tanh\left(\mathbf{W}_{\mathrm{init}}\bar{\mathbf{a}}\right) \quad (16)$$

where $\mathbf{W}_{\mathrm{init}} \in \mathbb{R}^{n\times D}$. By initializing GRU hidden state in this way, the parser is easier to train properly compared with initializing GRU hidden state as a zero-vector.

D. Guided hybrid attention

The proposed hybrid attention aims to teach the parser where to attend and when to attend. It consists of a coverage based spatial attention and a temporal attention. Fig. 4 shows the schematic representation of spatial and temporal attention. An attention guider that guides the learning of hybrid attention is embedded during the learning procedure.

1) Spatial attention: Intuitively, for each predicted symbol from the parser, the entire input sequence is not necessary to provide the useful information. Only a subset of adjacent trajectory points will mainly contribute to the computation of context vector $\mathbf{c}_t$ at each time step t. For example, in Fig. 2, the symbol "$\sum$" in the output sequence corresponds only to the red part in the purple rectangle: the other parts of the input expression do not need to be attended. Therefore, the parser can adopt a spatial attention mechanism to know where is the suitable place to attend to generate the next predicted symbol and then assign a higher weight to the corresponding local annotation vectors $\mathbf{a}_i$. Here, we parameterize the attention model as a multi-layer perceptron (MLP) which is jointly trained with the tracker and the parser:

$$e_{ti} = \boldsymbol{\nu}_{\mathrm{att}}^{\mathrm{T}}\tanh(\mathbf{W}_{\mathrm{att}}\hat{\mathbf{s}}_t+\mathbf{U}_{\mathrm{att}}\mathbf{a}_i) \quad (17)$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{L}\exp(e_{tk})} \quad (18)$$

where e_{ti} denotes the energy of annotation vector $\mathbf{a}_i$ at time step t conditioned on the current GRU hidden state prediction $\hat{\mathbf{s}}_t$, α_{ti} denotes the spatial attention coefficient of $\mathbf{a}_i$ at time step t. Let n' denote the attention dimension; then $\boldsymbol{\nu}_{\mathrm{att}} \in \mathbb{R}^{n'}$, $\mathbf{W}_{\mathrm{att}} \in \mathbb{R}^{n'\times n}$ and $\mathbf{U}_{\mathrm{att}} \in \mathbb{R}^{n'\times D}$. With the weights α_{ti}, we compute a context vector candidate $\hat{\mathbf{c}}_t$ as:

$$\hat{\mathbf{c}}_t = \sum_{i=1}^{L}\alpha_{ti}\mathbf{a}_i \quad (19)$$

We can understand the summation of all the annotations using weight coefficients as computing an expected annotation, which has a fixed-length 1 regardless of the variable-length of input traces.

2) Coverage model: There is one problem for the classic spatial attention mechanism in Eq. (17), namely lack of coverage [46], [52]. Coverage means the overall alignment information indicating whether a local region of the input traces has been parsed. The overall alignment information is especially important when recognizing HMEs because in principle, each part of input traces should be parsed only once. Lacking coverage will lead to misalignment resulting in over-parsing or under-parsing. Over-parsing implies that some parts of the input traces have been parsed twice or more, while under-parsing denotes that some parts have never been parsed. To address this problem, we append a coverage vector to the computation of attention in Eq. (17). The coverage vector aims at tracking the past alignment information. Different from [32], we compute the coverage vector based on the summation of all past attention probabilities, which can describe the alignment history:

$$\mathbf{F} = \mathbf{Q} * \sum_{l=1}^{t-1}\boldsymbol{\alpha}_l \quad (20)$$

$$e_{ti} = \boldsymbol{\nu}_{\mathrm{att}}^{\mathrm{T}}\tanh(\mathbf{W}_{\mathrm{att}}\hat{\mathbf{s}}_t+\mathbf{U}_{\mathrm{att}}\mathbf{a}_i+\mathbf{U}_f\mathbf{f}_i) \quad (21)$$

where $\boldsymbol{\alpha}_l$ denotes the attention probability vector at time step l, $\mathbf{Q}$ denotes a 1D convolution filter with q output channels and $\mathbf{f}_i$ denotes the i^{th} coverage vector of $\mathbf{F}$ 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 help adjust the future attention. More specifically, points in the input traces already significantly contributed to the generation of target symbols should be assigned with lower spatial attention probabilities in the following parsing phases. On the contrary, points with less contributions should be assigned with higher spatial attention probabilities. Consequently, the parsing process is finished only when the entire input traces have contributed and the problems of over-parsing or under-parsing can be alleviated.

3) Temporal attention: The coverage based spatial attention seems to be effective for generating the alignment between the target symbol and a local region of input traces. However, regarding to the criterion of generating LaTeX notations, there is one type of symbols, named as v-symbols (virtual symbols in LaTeX syntax for disambiguation without correspondences in math expressions), e.g., the symbol "{" in LaTeX caption of Fig. 2, which can be predicted reliably just based on the language model. The spatial attention often produces incorrect alignment when encountering these v-symbols. Furthermore, due to the ambiguities of handwriting input, the high-level representations extracted from trajectory information are sometimes unreliable. Therefore, we present a temporal attention to help the parser determine when to rely on the trajectory information and when to only rely on the language model.

By considering that the temporal attention should establish an adaptive gate to determine whether to attend to traces or strengthen the language model, we first design a supplementary vector as:

$$\mathbf{g}_t = \sigma(\mathbf{W}_{yg}\mathbf{y}_{t-1} + \mathbf{U}_{sg}\mathbf{s}_{t-1}) \tag{22}$$

$$\mathbf{m}_t = \mathbf{g}_t \otimes \tanh(\mathbf{W}_{\hat{s}}\hat{\mathbf{s}}_t) \tag{23}$$

where $\hat{\mathbf{s}}_t$ performs like a memory cell which stores both long and short term linguistic information as described in Eq. (12). So we reuse the memory cell to generate the supplementary vector $\mathbf{m}_t$ to strengthen the language model, $\mathbf{m}_t \in \mathbb{R}^D$. $\mathbf{g}_t$ denotes an update gate, $\mathbf{y}_{t-1}$ denotes the previous target symbol and $\mathbf{s}_{t-1}$ denotes the previous hidden state as in Eq. (12). $\mathbf{W}_{yg}$, $\mathbf{U}_{sg}$ and $\mathbf{W}_{\hat{s}}$ are related weight matrices.

Suppose the temporal attention should indicate how much attention the parser is placing on the language model (as opposed to the input traces), we compute it as follows:

$$\bar{e}_t = \frac{1}{L}\sum\nolimits_{i=1}^{L} e_{ti} \tag{24}$$

$$\mathbf{z}_t = [\bar{e}_t; \boldsymbol{\nu}_{\text{att2}}^{\mathrm{T}} \tanh(\mathbf{W}_{\text{att}}\hat{\mathbf{s}}_t + \mathbf{U}_m\mathbf{m}_t)] \tag{25}$$

$$\beta_t = \frac{\exp(\mathbf{z}_t[1])}{\exp(\mathbf{z}_t[0]) + \exp(\mathbf{z}_t[1])} \tag{26}$$

where e_{ti} is defined in Eq. (17), $\bar{e}_t$ is the average energy at time t, $\boldsymbol{\nu}_{\text{att2}} \in \mathbb{R}^{n'}$, $\mathbf{U}_m \in \mathbb{R}^{n' \times D}$, $\mathbf{W}_{\text{att}}$ is the same as in Eq. (17), and the temporal attention β_t is a scalar in the range $[0,\ 1]$.

As shown in Fig. 4, the context vector $\mathbf{c}_t$ is modeled as a mixture of the spatially attended annotation vector $\hat{\mathbf{c}}_t$ and the supplementary vector $\mathbf{m}_t$, which is calculated as:

$$\mathbf{c}_t = \beta_t\mathbf{m}_t + (1 - \beta_t)\hat{\mathbf{c}}_t \tag{27}$$

This formulation encourages the parser to adaptively attend to the annotations vs. the supplementary vector when generating the next symbol. The temporal attention scalar is updated at each time step.

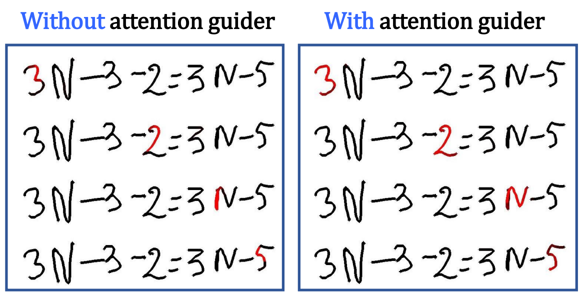


Fig. 5. Examples of attention maps with and without attention guider.

4) Attention guider: Although the attention mechanism achieves impressive results in machine translation, image captioning, speech recognition and even in OHMER, how to train it properly remains a challenging problem. While most previous studies train the attention model with random initialization, we believe the spatial attention should be tuned under a guider, as it plays such an important role in aligning the output strings with the input traces and controlling the flow of forward information and backward gradients.

Specific to OHMER, it is possible to extract the oracle alignment information from training data (e.g., in CROHME). Take expression "$x^2 - 1$" as an example, a writer may write down five strokes to represent it: the first two strokes for "x", the third stroke for "2", the fourth stroke for "$-$" and the last stroke for "1". So, if we want to generate the minus symbols "$-$", the oracle spatial alignment will attend to the fourth stroke and the spatial attention probability should be equally distributed only on the fourth stroke.

Concretely, we first consider the case when the ground truth spatial attention map $\gamma_t = \{\gamma_{ti}\}_{i=1,\ldots,L}$ is provided for the symbol w_t, with $\gamma_{ti} = \frac{1}{L}$ for each i. Note that $\sum_{i=1}^{L}\gamma_{ti} = \sum_{i=1}^{L}\alpha_{ti} = 1$, therefore they can be considered as two probability distributions of spatial attention and it is natural to employ the cross entropy function as the guider:

$$G_t = -\sum\nolimits_{i=1}^{L} \gamma_{ti} \log \alpha_{ti} \tag{28}$$

We add this spatial attention guider as a regularization term to the final objective function during training in Section V-A. As for symbols without explicit spatial alignment to input traces (e.g. "$\wedge$", "{"), we simply remove the guider. Fig. 5 shows the comparison between spatial attention with and without guider. For each predicted symbol, the traces highlighted by red color are the alignments introduced by spatial attention. It is clear that the spatial alignment learning with a guider is more reasonable.

IV. Illustration of Additional Models

A. Dynamic traces vs. static images

For OHMER, we observe that there is a strong complementarity between the dynamic-trace and static-image based representations. First, dynamic traces can provide additional rich dynamic information (i.e., the writing order information) over the static images, which can significantly improve the recognition accuracy. For example, handwritten math symbol

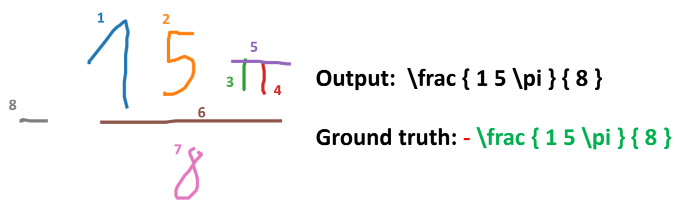

1
2
5
3
4
8
6
7
Output: \frac { 1 5 \pi } { 8 }
Ground truth: - \frac { 1 5 \pi } { 8 }

Fig. 6. An incorrectly recognized example due to the delayed stroke.

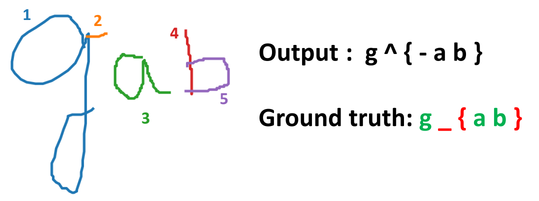


Fig. 7. An incorrectly recognized example due to the inserted stroke.

pairs "a" and "α", "b" and "6" are hard to distinguish in the form of static images due to the ambiguities of handwriting input and the similar character shape. However, the writing orders of each pair are totally different, which is one important discriminative feature.

Second, static-image based representation also possess its distinct advantage compared with dynamic traces as it can alleviate the under-parsing and over-parsing problems caused by delayed and inserted strokes. When using handwriting as input, the delayed and inserted strokes can be observed quite frequently. Delayed strokes occur as an extension of a fraction bar or a square root or transforming a symbol in another one by adding an additional stroke (e.g., "-" transformed into "+"). Fig. 6 shows an example of under-parsing problem, where the first symbol of LaTeX string (namely the minus sign "$-$" corresponding to the last handwritten stroke) is missing after OHMER using TAP. Normally we write the minus sign as the first stroke. Consequently, this delayed stroke leads to the under-parsing problem. Inserted strokes occur as splitting a successive stroke into several broken strokes by anomalous handwriting or extra meaningless strokes. In Fig. 7, the second stroke, which is actually the end of symbol "g" but split into another meaningless stroke by the writer, is an inserted stroke which causes the over-parsing problem. Consequently, our model over-translates the input expression and recognizes the second stroke as a minus sign "$-$". Meanwhile, as the inserted stroke is on the up-right direction of basic symbol "g", the subscript structure is inaccurately recognized as the superscript structure. However, since the static images only focus on the character shape of handwritten symbols, both under-parsing and over-parsing problems can be well addressed.

B. Watch, Attend and Parse (WAP)

To fully utilize the advantages of dynamic traces and static images simultaneously, we propose to blend WAP approach in [46] with TAP. We first draw the trajectory, remove the writing order information and transform the dynamic traces into static images as described in Fig. 1. Then, we employ a FCN based watcher to map HME images to high-level features. Finally, a GRU based parser converts these high-level features into output LaTeX strings, symbol by symbol. More implementation details about WAP can be found in our previous work [46]. The integration of WAP and TAP models in the recognition stage is elaborated in Section V-B.

C. Language model (LM)

Statistical language models are crucial to many applications, such as automatic speech recognition and statistical machine translation. Compared with conventional methods (e.g. the popular N-grams [53]), language models based on recurrent neural networks achieve better performance [54]. In this study, we propose the GRU-based language model for OHMER to predict the next symbol in textual data given context. By feeding the officially provided text data containing 173500 LaTeX notations of mathematical expressions into a single unidirectional GRU with cross-entropy as optimization function, we train a new GRU-based language model, which is supposed to be stronger than the implicit language model built in the parser of TAP trained with 8836 HMEs. In Section V-B, the integration with GRU-based language model in the decoding process is introduced.

V. Training and Decoding Procedure

A. Training

The training objective of our model is to maximize the predicted symbol probability as shown in Eq. (11) and we use cross-entropy (CE) function as the cost. The objective function for optimization, which consists of the CE cost and the attention guider, is shown as follows:

$$O = -\sum\nolimits_{t=1}^{C} \log p(w_t|\mathbf{y}_{t-1}, \mathbf{X}) + \lambda \sum\nolimits_{t=1}^{C} G_t \qquad (29)$$

where w_t represents the ground truth word at time step t, C is the length of output string, G_t is the attention guider, and λ is set to 0.1.

The tracker consists of 4 bidirectional GRU layers. Each layer has 250 forward and 250 backward GRU units. The pooling is applied to the top 2 GRU layers over time. Accordingly, the tracker reduces the input sequence length by the factor of 4. The parser adopts 2 unidirectional GRU layers and each layer has 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. The kernel size of convolution filter $\mathbf{Q}$ for computing coverage vector is set to (121×1) and the number of output channels q is 256. We utilize the adadelta algorithm [55] 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) of validation set. We use the weight noise [56] as the regularization. The first-pass training is conducted without weight noise. Then we anneal the best model in terms of WER by restarting the training with weight noise.

B. Decoding

In the decoding stage, we aim to generate a most likely LaTeX string given the input HME traces.

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \log P\left(\mathbf{y}|\mathbf{x}\right) \tag{30}$$

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 [57] is employed to implement the decoding procedure. Here, we maintain a set of 10 partial hypotheses, beginning with the start-of-sentence token $< sos >$. At each time step, each partial hypothesis in the beam is expanded with every generated symbol and only the hypotheses with 10 minimal scores are retained:

$$\mathbf{S}_t = \mathbf{S}_{t-1} - \log p(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{x}) \tag{31}$$

where $\mathbf{S}_t$ represents the score at time step t, $p(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{x})$ represents the probability of all generated symbols in the dictionary. This procedure is repeated until the output symbol becomes the end-of-sentence token $< eos >$.

During the beam search procedure, it is intuitive to adopt the ensemble method [58] for improving the performance. We first train N_1 TAP models on the same training set but with different initialized parameters. Then we can average their prediction probabilities $p_1^i(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{x})$ to predict the current output symbol:

$$\mathbf{S}_t = \mathbf{S}_{t-1} - \log\left(\frac{1}{N_1}\sum\nolimits_{i=1}^{N_1} p_1^i(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{x})\right) \tag{32}$$

As mentioned in Section IV, we also generate the prediction probability $p_2^i(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{x})$ based on WAP model and the prediction probability $p_3^i(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{x})$ based on a stronger GRU-based language model. We blend TAP with WAP and the additional GRU-based language model in the beam search procedure as:

$$\begin{aligned}\mathbf{S}_t = \mathbf{S}_{t-1} - \log(&\xi_1 \times \frac{1}{N_1}\sum\nolimits_{i=1}^{N_1} p_1^i(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{x})\\ &+\xi_2 \times \frac{1}{N_2}\sum\nolimits_{i=1}^{N_2} p_2^i(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{x})\\ &+\xi_3 \times \frac{1}{N_3}\sum\nolimits_{i=1}^{N_3} p_3^i(\mathbf{y}_t|\mathbf{y}_{t-1}, \mathbf{x}))\end{aligned} \tag{33}$$

where ξ_1, ξ_2 and ξ_3 denote the ratio of contribution, $\xi_1 + \xi_2 + \xi_3 = 1$, N_1, N_2 and N_3 denote the number of their respective ensemble models, we set $N_1 = N_2 = N_3 = 3$.

VI. Experiments

We design a set of experiments to validate the effectiveness of the proposed method for OHMER by answering the following questions:

Q1 Is the proposed guided hybrid attention effective?
Q2 How does TAP analyze the 2D structure of HME?
Q3 Can WAP and GRU-LM yield additional gains?
Q4 Does the proposed approach outperform others?

The experiments are all implemented with Theano 0.10.0 [59] and an NVIDIA Tesla M40 12G GPU. And our source code is publicly available [1].

[1]https://github.com/JianshuZhang/TAP

A. Dataset and metric

The experiments are conducted on CROHME competition dataset [60], [61], [62], which is currently the most widely used public dataset for OHMER. The CROHME 2014 dataset has a training set of 8836 HMEs (86K symbols) and a test set of 986 HMEs (6K symbols). There are 101 math symbol classes. None of the handwritten expressions or LaTeX notations in the test set appears in the training set. To be fairly comparable, we use the CROHME 2013 test set as a validation set for estimating models during the training, just like other participants of CROHME 2014 competition. As for the CROHME 2016 competition, the training set is the same as CROHME 2014. But the test set is newly collected and labeled by the organizers at the University of Nantes. There are totally 1147 expressions and the symbol classes remain unchanged.

The participating systems in all of the CROHME competitions are 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 transfer the generated LaTeX strings into MathML representation and then compute 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 the expression level. Here, we also evaluate our system at the symbol-level by using WER [63] metric, which reveals errors such as substitutions, deletions and insertions.

B. Evaluation of guided hybrid attention (Q1)

In this section, we show the effectiveness of each component in guided hybrid attention through several designed systems in Table I.

TABLE I
Comparison among systems from P1 to P8. Attributes for comparison include: 1) employing the classic spatial attention model; 2) appending a coverage vector into the classic spatial attention model; 3) employing the temporal attention model; 4) employing the attention guider; 5) using ensemble method as described in Eq. (32).

System	Spatial	Coverage	Guider	Temporal	Ensemble
P1	✓	-	-	-	-
P2	✓	✓	-	-	-
P3	✓	✓	✓	-	-
P4	✓	✓	✓	✓	-
P5	✓	-	-	-	✓
P6	✓	✓	-	-	✓
P7	✓	✓	✓	-	✓
P8	✓	✓	✓	✓	✓

The classic spatial attention model is essential for each system. Meanwhile, the temporal attention only works well when the attention guider is also implemented, which might be explained as that the small training set is not adequate for the hybrid attention to train the model parameters properly with random initialization.

TABLE II
COMPARISON OF RECOGNITION PERFORMANCE (IN %) AND TIME EFFICIENCY (IN SECOND) AMONG DIFFERENT SYSTEMS IN TABLE I ON CROHME 2014. TRAIN TIME DENOTES THE TIME COST FOR ONLY ONE EPOCH, EPOCHS DENOTES THE NUMBER OF NEEDED EPOCHES FOR TRAINING, TEST SPEED DENOTES THE TIME COST FOR EVALUATION ON THE WHOLE CROHME 2014 TEST SET (986 HMES). TRAIN TIME OF ENSEMBLE SYSTEMS **P5-P8** IS NOT SHOWN AS THEIR BASE MODELS HAVE BEEN ALREADY TRAINED.

System	WER	ExpRate	Train Time (Epochs)	Test Speed
P1	19.33	42.49	476 (157)	70
P2	16.56	46.86	710 (166)	118
P3	14.17	49.29	775 (149)	116
P4	13.39	50.41	780 (185)	115
P5	14.86	48.38	-	214
P6	12.64	52.43	-	380
P7	11.91	54.36	-	378
P8	11.53	55.37	-	377

In Table II, we show the recognition performance and time efficiency of systems P1-P8 on CROHME 2014 test set. Firstly, by the comparison of ExpRate we show the improvement via coverage vector, attention guider and temporal attention by appending each of them to their previous system step by step. More specifically, the ExpRate is increased from 42.49% to 46.86% after the coverage vector is appended into the classic spatial attention model (P1 vs. P2). As already illustrated in Fig. 5, the attention guider drives the attention alignment more reasonable and we prove that more reasonable attention can lead to better recognition performance as the ExpRate is increased from 46.86% to 49.29% after the attention guider is equipped (P2 vs. P3). By comparing P3 with P4, the proposed temporal attention could also improve the ExpRate from 49.29% to 50.41%. To better interpret its superiority, we show an example of OHMER improved by temporal attention in Fig. 8. Beta represents the temporal attention gate in Eq. (27), the higher the parser should not pay attention to input traces. P4 successfully learns to less attend to the input traces when generating v-symbols. Furthermore, when encountering ambiguous symbols such as the symbol in red rectangle, P4 also assigns a relatively high temporal attention probability. By more attending to the language model and less attending to input traces, P4 successfully recognize it as “f” rather than “8”, since the case of the symbol “e” followed by the symbol “8” unlikely happens in math language.

Secondly, we show the comparison of computational cost among P1-P4 by investigating their Train Time and Test Speed. As we can see in Table II, system P1 needs 476 seconds for one epoch. The overall training procedure takes 21 hours for P1 to converge as it needs nearly 157 epochs. During testing, we evaluate 986 HMEs of CROHME 2014 test set one by one and P1 costs 70 seconds to finish the testing procedure. Comparing P1 with P2, the coverage based attention model slows down the training procedure and testing procedure even if it brings a great improvement on ExpRate. After appending the attention guider (P2 vs. P3), our model becomes easier to train properly (need less epochs). Since it performs as

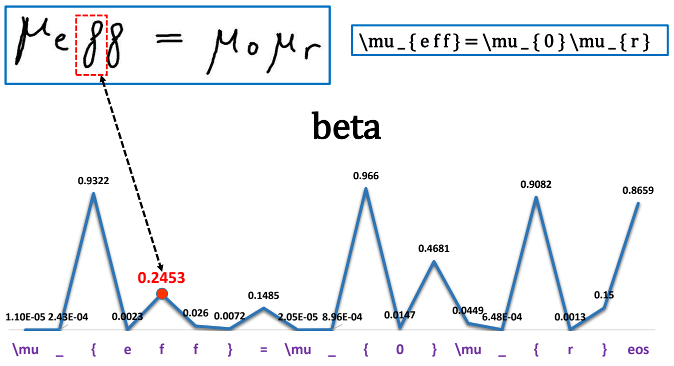


Fig. 8. An illustration of using temporal attention, symbol in the red rectangle is incorrectly recognized as “8” without temporal attention and correctly recognized as “f” with temporal attention.

a regularization term during training, it does not affect the Test Speed but it slows down the backward propagation of gradient during training. The computational cost of temporal attention can be ignored (P3 vs. P4) but it takes more epochs to converge. The Test Speed of P4 is even slightly faster than P3 as the recognition performance of P4 is better, encountering less insertion error during decoding.

Finally, we compare the ensemble systems P5-P8 with single systems P1-P4 respectively. It is obvious to see that combining 3 models in an ensemble way (Eq. (32)) can yield more than 5% absolute gain but also brings 3 times computational cost for testing.

C. Attention visualization (Q2)

In this section, we show through attention visualization how the proposed model is able to analyze the two-dimensional structure of HME. The proposed TAP approach to perform symbol segmentation implicitly is also explained. We draw the trajectory of input HME in a 2D 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.

To analyze the 2D structure of HME, it is essential to identify the spatial relationships between math symbols, which are statistically determined and might be horizontal, vertical, subscript, superscript or inside. As shown in Fig. 9, the horizontal and vertical relationships are easy to learn by focusing on the middle operator. To handle the superscripts, the parser precisely pays attention to the end of base symbols and the start of superscript symbols, which is reasonable because traces in the start of superscript symbols are on the upper-right of traces in the end of base symbols, describing the upper-right direction. Similarly, for subscripts, the ending traces of base symbols and the starting traces of subscript symbols can also describe a bottom-right direction. As for the inside relationships, the decoder attends to the bounding symbols.

More specifically, in Fig. 10, 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

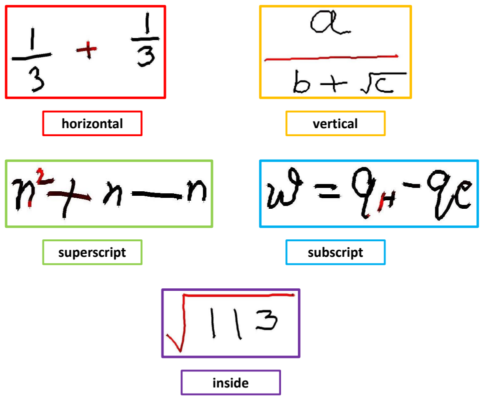


Fig. 9. Learning of five spatial relationships (horizontal, vertical, subscript, superscript and inside) through attention visualization.

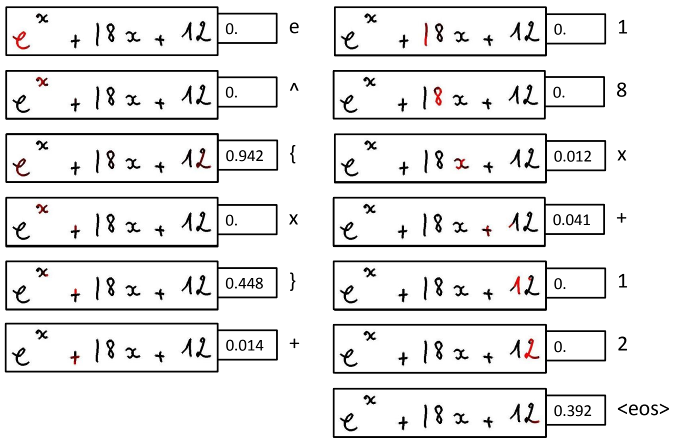


Fig. 10. Hybrid attention visualization for an example of an HME with the LaTeX ground truth "e ∧ { x } + 1 8 x + 1 2", spatial attention is shown through red color in images and temporal attention is shown in the attached boxes, on the right of boxes are predicted symbols.

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 distinguishes the upper-right direction and then produces the symbol "∧". 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 and the temporal attention coefficients increase significantly when encountering these v-symbols ("{}").

D. Evaluation of model combination (Q3)

In Table III, we show the improvements and additional computational cost via a WAP model and an additional GRU-based language model by appending each of them to the proposed TAP model step by step. Here, Time denotes the total seconds for each system to finish the evaluation of CROHME 2014 (986 HMEs) and CROHME 2016 test set (1147 HMEs).

S1 denotes a pure WAP model that takes only static expression images as input. S2 still denotes the WAP model but

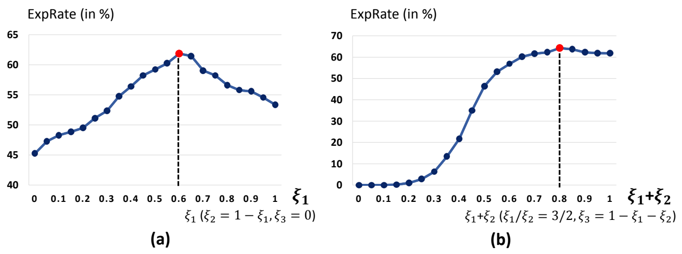


Fig. 11. (a) The curve of ExpRate with respect to ξ_1 (the contribution of TAP) in the ensemble of TAP and WAP (with $\xi_2 = 1 - \xi_1, \xi_3 = 0$); (b) The curve of ExpRate with respect to $(\xi_1 + \xi_2)$ in the ensemble of (TAP+WAP) and GRU-LM (with $\xi_1/\xi_2 = 3/2, \xi_3 = 1 - \xi_1 - \xi_2$). We draw the two curves on the validation set.

the input contains static expression images and additional 8-directional pattern images. The 8-directional pattern images are the intermediate products of the 8-directional raw features [64] which contain dynamic trajectory information. The dynamic trajectory information embedded in 8-directional pattern images helps WAP achieve a 2% gain (S1 vs. S2) and the computational cost can be ignored. S3 denotes the proposed TAP model which is same one as P8 in Table II and its performance shows the great superiority of dynamic trajectory information for OHMER compared with static images. Then, we can observe that the ExpRate increases about 5% compared to TAP after the combination of TAP and WAP (S4), which demonstrates the strong complementarity between dynamic traces and static images. The optimal weighting coefficients ξ_1 and ξ_2 for TAP and WAP can be determined by Fig. 11(a) on the validation set. Note that S2 also utilizes the complementarity between traces and images, but the gap between S2 and S4 indicates that RNN seems a better way for processing dynamic trajectory information even if it takes nearly twice time than CNN for evaluation. Furthermore, after we blend the GRU-based language model into the ensemble of TAP and WAP (S5), the ExpRate increases from 60.34% to 61.16% on CROHME 2014, and from 55.27% to 57.02% on CROHME 2016. As the language model is only a single unidirectional GRU layer, it does not bring much time consumption. The optimal weighting coefficients can be determined by Fig. 11(b), namely $\xi_1 = 0.48, \xi_2 = 0.32, \xi_3 = 0.2$.

TABLE III
COMPARISON OF RECOGNITION PERFORMANCE (IN %) AND TIME EFFICIENCY (IN SECOND) AMONG FIVE DIFFERENT SYSTEMS ON CROHME 2014 AND CROHME 2016. **S1** DENOTES SYSTEM WAP, **S2** DENOTES SYSTEM WAP USING 8-DIRECTIONAL TRAJECTORY FEATURES, **S3** DENOTES SYSTEM TAP, **S4** DENOTES SYSTEM TAP COMBINING WAP, **S5** DENOTES SYSTEM TAP COMBINING WAP AND GRU-BASED LANGUAGE MODEL.

System	CROHME 2014			CROHME 2016		
	WER	ExpRate	Time	WER	ExpRate	Time
S1	19.40	44.42	198	19.73	42.02	233
S2	17.73	46.55	196	16.88	44.55	230
S3	11.53	55.37	377	12.62	50.22	455
S4	9.95	60.34	524	10.59	55.27	712
S5	9.73	61.16	564	10.55	57.02	745

E. Comparison with state-of-the-arts (Q4)

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

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
Ours	**61.16**	**75.46**	**77.69**	**78.19**

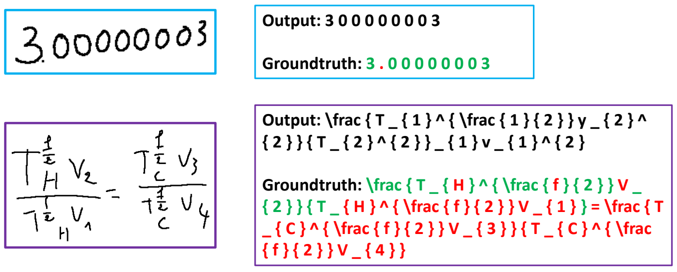
First, we make a comparison of our best system and other submitted systems on both CROHME 2014, as shown in Table IV. Details of these systems can refer to [61]. To make a fair comparison among different systems, we only list the results using the CROHME training set. Our model represents the ensemble of three TAP models, three WAP models and three GRU-based language models. The textual data for training the language model is also provided by CROHME 2014. There was a large gap between the top-1 system of CROHME 2014 competition and our proposed system. Although another system named "MyScript" of CROHME 2014 competition achieved a slightly higher result (with an ExpRate of 62.68%), that system used a large private dataset and the technical details were unrevealed.

TABLE V
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	49.61	60.42	64.69	–
Tokyo	43.94	50.91	53.70	–
São Paolo	33.39	43.50	49.17	–
Nantes	13.34	21.02	28.33	–
Ours	**57.02**	**72.28**	**75.59**	**76.19**

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 V, with an ExpRate of 57.02% which was quite a promising result compared with other participating systems. The team Wiris was awarded the first place on CROHME 2016 competition using only the CROHME training data with an ExpRate of 49.61%, and it used a Wikipedia formula corpus, consisting of more than 592000 formulae, to train a strong language model. The details of other systems refer to [62].

Note that, CROHME participants evaluate their predicted HMEs based on LG representations which identify the labels of each stroke. However, in our experiments we evaluate predicted HMEs based on MathML representations. Compared with evaluation based on LG representations, the MathML evaluation is less strict as it does not check the stroke segmentation. We adopt MathML representations rather than LG representations because our system only aims to directly generate the predicted LaTeX string, it does not provide accurate



Output: 3 0 0 0 0 0 0 0 3

Groundtruth: 3 . 0 0 0 0 0 0 0 3

Output: \frac { T _ { 1 } ^ { \frac { 1 } { 2 } } y _ { 2 } ^ { 2 } } { T _ { 2 } ^ { 2 } } _ { 1 } v _ { 1 } ^ { 2 }

Groundtruth: \frac { T _ { H } ^ { \frac { f } { 2 } } V _ { 2 } } { T _ { H } ^ { \frac { f } { 2 } } V _ { 1 } } = \frac { T _ { C } ^ { \frac { f } { 2 } } V _ { 3 } } { T _ { C } ^ { \frac { f } { 2 } } V _ { 4 } }

Fig. 12. Two examples of HME which are incorrectly recognized, besides the HME images are their predicted output and ground truth, in the ground truth green texts are predicted correctly while red texts are predicted incorrectly.

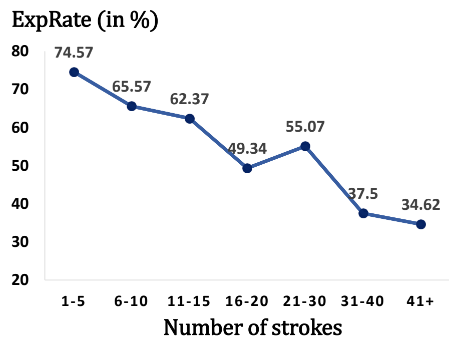


Fig. 13. Number of strokes vs. ExpRate (in %) on CROHME 2014.

stroke segmentation results. Although such issue might not change much the recognition performance, we will improve our system to provide stroke-level evaluation by refining the alignment produced from attention model in the future so that our performance are completely comparable with CROHME participants.

F. Error analysis

Although WAP and GRU-LM can help alleviate some over-parsing and under-parsing problems coming from TAP, the combination of TAP, WAP and GRU-LM still reveals some problems need to be addressed. Fig. 12 shows two examples of HME that are incorrectly recognized. The blue one is an under-parsing problem as the decimal point "." is missed in the predicted LaTeX string. As we can see the decimal point is very close to math symbol "3" and its scale is much smaller than its adjacent symbols. After some pooling operations (subsampling in TAP or max-pooling in WAP) that will drop the fine-grained details of extracted features, the visual information of the decimal point is gone, leading to an under-parsing problem. The purple example in Fig. 12 shows a disastrous error that most parts of HME are missing in the predicted LaTeX string, even if the simple horizontal structure (represented by math symbol "=") has not been parsed. The diaster happens as the parser performs more like a language model without considering the HME as a composition of several sub-expressions, therefore previous predicted errors are accumulated severely and inherited by next decoding step.

Another interesting analysis concerns the distribution of ExpRate with respect to the number of strokes. Fig. 13 illustrates this behaviour. In general, when the number of strokes is no more than 15, HMEs are inaccurately recognized due to failure of symbol recognition, mostly resulting from handwriting ambiguities and similarities among math symbols. We expect the model to perform poorly on HMEs composed of large number of strokes due to 1) the number of training samples composed of many strokes is quite limited; 2) HMEs that consist of many strokes are usually related with long LaTeX strings and HMEs with longer LaTeX strings are more likely to bring about the mentioned disastrous errors.

VII. Conclusion

In this study we introduce an end-to-end framework with guided hybrid attention based model to recognize online handwritten mathematical expressions. The proposed model is data-driven and alleviates the problem of explicit segmentation. We demonstrate through visualization and experiment results that the novel guide hybrid attention performs better than the conventional attention model. We also verify that there is a strong complementarity between static-image based representation and dynamic-trace based representation for OHMER. Combining TAP with WAP and an additional language model, we achieve promising recognition results on both CROHME 2014 and CROHME 2016 competition.

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, and MOE-Microsoft Key Laboratory of USTC.

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] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] T. H. Rhee and J. H. Kim, "Efficient search strategy in structural analysis for handwritten mathematical expression recognition," *Pattern Recognition*, vol. 42, no. 12, pp. 3192–3201, 2009.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

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

[14] S. Lavirotte and L. Pottier, "Mathematical formula recognition using graph grammar," in *Electronic Imaging*, vol. 3305, 1998.

[15] Y. Eto and M. Suzuki, "Mathematical formula recognition using virtual link network," in *Document Analysis and Recognition, 2001. Proceedings. Sixth International Conference on*. IEEE, 2001, pp. 762–767.

[16] L. Hu and R. Zanibbi, "MST-based visual parsing of online handwritten mathematical expressions," in *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*. IEEE, 2016, pp. 337–342.

[17] K. Cho, A. Courville, and Y. Bengio, "Describing multimedia content using attention-based encoder-decoder networks," *IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 1875–1886, 2015.

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

[19] 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.

[20] L. Lamport, *LATEX: a document preparation system: user's guide and reference manual*. Addison-wesley, 1994.

[21] 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.

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

[23] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[24] 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.

[25] 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.

[26] K.-F. Chan and D.-Y. Yeung, "Error detection, error correction and performance evaluation in on-line mathematical expression recognition," *Pattern Recognition*, vol. 34, no. 8, pp. 1671–1684, 2001.

[27] R. Yamamoto, S. Sako, T. Nishimoto, and S. Sagayama, "On-line recognition of handwritten mathematical expressions based on stroke-based stochastic context-free grammar," in *Tenth International Workshop on Frontiers in Handwriting Recognition*, 2006.

[28] S. MacLean and G. Labahn, "A new approach for recognizing handwritten mathematics using relational grammars and fuzzy sets," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 16, no. 2, pp. 139–163, 2013.

[29] 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.

[30] M.-T. Luong, I. Sutskever, Q. V. Le, O. Vinyals, and W. Zaremba, "Addressing the rare word problem in neural machine translation," *arXiv preprint arXiv:1410.8206*, 2014.

[31] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell," *arXiv preprint arXiv:1508.01211*, 2015.

[32] 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.

[33] 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.

[34] ——, "Trajectory-based radical analysis network for online handwritten chinese character recognition," *arXiv preprint arXiv:1801.10109*, 2018.
[35] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3156–3164.
[36] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3128–3137.
[37] L. Gao, Z. Guo, H. Zhang, X. Xu, and H. T. Shen, "Video captioning with attention-based LSTM and semantic consistency," *IEEE Transactions on Multimedia*, vol. 19, no. 9, pp. 2045–2055, 2017.
[38] N. Zhao, H. Zhang, R. Hong, M. Wang, and T.-S. Chua, "Videowhisper: Toward discriminative unsupervised video feature learning with attention-based recurrent neural networks," *IEEE Transactions on Multimedia*, vol. 19, no. 9, pp. 2080–2092, 2017.
[39] R. A. Rensink, "The dynamic representation of scenes," *Visual cognition*, vol. 7, no. 1-3, pp. 17–42, 2000.
[40] M. Corbetta and G. L. Shulman, "Control of goal-directed and stimulus-driven attention in the brain," *Nature reviews. Neuroscience*, vol. 3, no. 3, p. 201, 2002.
[41] G. Evangelopoulos, A. Zlatintsi, A. Potamianos, P. Maragos, K. Rapantzikos, G. Skoumas, and Y. Avrithis, "Multimodal saliency and fusion for movie summarization based on aural, visual, and textual attention," *IEEE Transactions on Multimedia*, vol. 15, no. 7, pp. 1553–1568, 2013.
[42] 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.
[43] C. Liu, J. Mao, F. Sha, and A. L. Yuille, "Attention correctness in neural image captioning." in *AAAI*, 2017, pp. 4176–4182.
[44] J. Lu, C. Xiong, D. Parikh, and R. Socher, "Knowing when to look: Adaptive attention via a visual sentinel for image captioning," *arXiv preprint arXiv:1612.01887*, 2016.
[45] L. Li, S. Tang, Y. Zhang, L. Deng, and Q. Tian, "Gla: Global-local attention for image description," *IEEE Transactions on Multimedia*, 2017.
[46] 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.
[47] Y. Deng, A. Kanervisto, J. Ling, and A. M. Rush, "Image-to-markup generation with coarse-to-fine attention," in *International Conference on Machine Learning*, 2017, pp. 980–989.
[48] J. Zhang, J. Du, and L. Dai, "A gru-based encoder-decoder approach with attention for online handwritten mathematical expression recognition," in *ICDAR 2017*, in press.
[49] 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*, 2017.
[50] 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.
[51] J. Zhang, J. Tang, and L.-R. Dai, "RNN-BLSTM based multi-pitch estimation." in *INTERSPEECH*, 2016, pp. 1785–1789.
[52] Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li, "Modeling coverage for neural machine translation," *arXiv preprint arXiv:1601.04811*, 2016.
[53] M. Damashek, "Gauging similarity with n-grams: Language-independent categorization of text," *Science*, vol. 267, no. 5199, p. 843, 1995.
[54] T. Mikolov, "Statistical language models based on neural networks," *Presentation at Google, Mountain View, 2nd April*, 2012.
[55] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.
[56] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems*, 2011, pp. 2348–2356.
[57] K. Cho, "Natural language understanding with distributed representation," *arXiv preprint arXiv:1511.07916*, 2015.
[58] T. G. Dietterich *et al.*, "Ensemble methods in machine learning," *Multiple classifier systems*, vol. 1857, pp. 1–15, 2000.
[59] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: A cpu and gpu math compiler in python," in *Proc. 9th Python in Science Conf*, 2010, pp. 1–7.
[60] 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.
[61] 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.
[62] 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.
[63] D. Klakow and J. Peters, "Testing the correlation of word error rate and perplexity," *Speech Communication*, vol. 38, no. 1, pp. 19–28, 2002.
[64] Z.-L. Bai and Q. Huo, "A study on the use of 8-directional features for online handwritten chinese character recognition," in *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*. IEEE, 2005, pp. 262–266.

Jianshu Zhang received the B.Eng. degree from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC) in 2015. He is currently a Ph.D. candidate of USTC. His current research area is neural network, handwriting mathematical expression recognition and Chinese document analysis.

Jun Du received the B.Eng. and Ph.D. degrees from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC) in 2004 and 2009, respectively. From 2004 to 2009, he was with iFlytek Speech Lab of USTC. During the above period, he worked as an Intern twice for 9 months at Microsoft Research Asia (MSRA), Beijing. In 2007, he also worked as a Research Assistant for 6 months in the Department of Computer Science, The University of Hong Kong. From July 2009 to June 2010, he worked at iFlytek Research on speech recognition. From July 2010 to January 2013, he joined MSRA as an Associate Researcher, working on handwriting recognition, OCR, and speech recognition. Since February 2013, he has been with the National Engineering Laboratory for Speech and Language Information Processing (NEL-SLIP) of USTC.

Lirong Dai was born in China in 1962. He received the B.S. degree in electrical engineering from Xidian University, Xian, China, in 1983, and the M.S. degree from the Hefei University of Technology, Hefei, China, in 1986, and the Ph.D. degree in signal and information processing from the University of Science and Technology of China (USTC), Hefei, in 1997. He joined USTC in 1993. He is currently a Professor at the School of Information Science and Technology, USTC. His research interests include speech synthesis, speaker and language recognition, speech recognition, digital signal processing, voice search technology, machine learning, and pattern recognition. He has published more than 50 papers in these areas.