Tree-based BLSTM for mathematical expression recognition

Ting Zhang, Harold Mouchère, Christian Viard-Gaudin

Tree-based BLSTM for mathematical expression recognition

Ting ZHANG, Harold MOUCHERE, Christian VIARD-GAUDIN
LS2N/IPI - UMR CNRS 6004, Université de Nantes, France
ting.zhang1@etu.univ-nantes.fr, {harold.mouchere, christian.viard-gaudin}@univ-nantes.fr

***Abstract*—In this study, we extend the chain-structured BLSTM to tree structure topology and apply this new network model for online math expression recognition. The proposed system addresses the recognition task as a graph building problem. The input expression is a sequence of strokes from which an intermediate graph is derived using temporal and spatial relations among strokes. In this graph, a node corresponds to a stroke and an edge denotes the relationship between a pair of strokes. Then several trees are derived from the graph and labeled with Tree-based BLSTM. The last step is to merge these labeled trees to build an admissible label graph (LG) modeling 2-D formulas uniquely. The proposed system achieves competitive results in online math expression recognition domain.**

I. Introduction

Handwritten mathematical expression (ME) recognition is an appealing topic in pattern recognition field since it exhibits a big research challenge and underpins many practical applications. From a scientific point of view, both a large set of symbols (more than 100) and 2 dimensional (2-D) structures increase the difficulty of this recognition problem; with regard to the application, it offers an easy and direct way to input MEs into computers, and therefore improves productivity for scientific writers. We usually divide handwritten MEs into online and offline domains. In the offline domain, data is available as an image, while in the online domain it is a sequence of strokes, which are themselves sequences of points recorded along the pen trajectory. Compared to the offline ME, time information is available in online form. This paper will be focused on online handwritten ME recognition.

Three tasks are involved in ME recognition [2], [1]: (1) symbol segmentation, which consists in grouping strokes that belong to the same symbol; (2) symbol recognition, the task of labeling the symbols to assign each of them a symbol class; (3) structural analysis, its goal is to identify spatial relations between symbols and with the help of a grammar to produce a mathematical interpretation. The state of the art solutions [3], [4], [1] are mainly grammar-driven solutions: a set of symbol hypotheses maybe generated and a structural analysis algorithm (grammar parsing usually) may select the best hypotheses while building the structure. However, these classical grammar-driven solutions, require not only a large amount of manual work for defining grammars, but also a high computational complexity for grammar parsing process.

In this work, we have been exploring online handwritten ME recognition from a new perspective, treating it as a problem of building a graph, and then, extracting several trees which are embbeded in the graph and recognized them with a tree-based BLSTM. As known, it is possible to describe a ME using a primitive Label Graph (LG, refer to Section II-A) in which nodes represent strokes whereas labels on the edges encode either segmentation or layout information. In [5], [18], the solution of building a graph by merging multiple 1D sequences of labels produced by a sequence labeler was proposed. This study will be focused on integrating multiple trees labeled by a tree labeler to recognize MEs.

Advanced recurrent neural network — Long Short-Term Memory (LSTM) — achieved great success in temporal dependency modeling for chain-structured data, such as text and speeches [6], [7]. This success is due to LSTM's representational power and effectiveness at capturing long-term dependency in a sequence. Recently, research on LSTM has been beyond sequential structure. It was extended to tree structure in [8], [9] and DAG (directed acyclic graph) structure in [10]. In this work, we put efforts on a similar task, generalizing the classical LSTM architecture to a tree network topology. Thereby, the new topology could handle tree-structured data as well as sequence-structured data.

In Section II, the related works will be reviewed, consisting of ME representation and an overview of LSTM. Tree-based BLSTM, as the base of our recognition system, is described in detail in Section III. Afterwards, we introduce our tree-based BLSTM recognition system in Section IV step by step, which is the main part of our study. At last, experiments and conclusion are covered in Sections V and VI respectively.

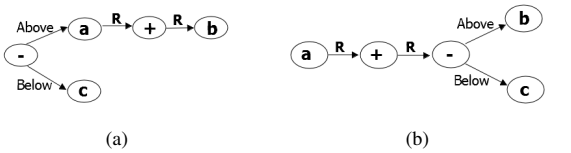
II. Related Work

A. ME Representation

Structures can be depicted at three different levels: symbolic, object and primitive [13]. In the case of handwritten ME, the corresponding levels are expression, symbol and stroke.

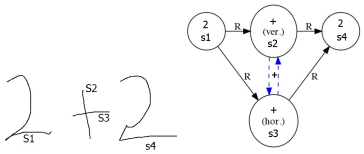
Symbol Relation Tree (SRT) It is possible to describe a ME at the symbol level using a SRT which represents spatial relationships between symbols. In SRT, nodes represent symbols, while labels on the edges indicate the relationships between symbols. For example, in Figure 1a, the symbol '-' on the base line is the root of the tree with symbol 'a' above and symbol 'c' below it. In Figure 1b, the symbol 'a' is the root; the symbol '+' is on the right of 'a'.

2379-2140/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDAR.2017.154



(a) (b)

Fig. 1. The symbol relation tree (SRT) for (a) $\frac{a+b}{c}$ and (b) $a + \frac{b}{c}$. 'R' is for left-right relationship



(a) (b)

Fig. 2. (a) '2 + 2' written with four strokes; (b) the SLG of '2 + 2'. The four strokes are indicated as s1, s2, s3, s4 in writing order. (ver.) and (hor.) are added to differentiate the vertical and the horizontal strokes for '+'. 'R' is for left-right relationship

Label Graph (LG) If we go down at the stroke level, a LG, and more specifically a Stroke Label Graph (SLG), can be derived from the SRT. In LG, nodes represent strokes, while labels on the edges encode either segmentation information or symbol relationships. Consider the simple expression '2 + 2' written using four strokes (two strokes for '+'), the handwritten input and its LG can be seen in Figure 2a and 2b. As illustrated, nodes of SLG are labeled with the class of the corresponding symbol to which the stroke belongs. A dashed edge corresponds to segmentation information; it indicates that a pair of strokes belongs to the same symbol. In this case, the edge label is the same as the common symbol label. On the other hand, the non-dashed edges define spatial relationships between nodes and are labeled with one of the different possible relationships between symbols. As a consequence, all strokes belonging to the same symbol are fully connected, nodes and edges sharing the same symbol label; when two symbols are in relation, all strokes from the source symbol are connected to all strokes from the target symbol by edges sharing the same relationship label. The spatial relationships as defined in the CROHME [1] competition are: $Right$, $Above$, $Below$, $Inside$ (for square root), $Superscript$, $Subscript$.

B. Long Short-Term Memory Networks

Recurrent neural networks (RNNs). RNNs can access contextual information and therefore are suitable for sequence labeling [11]. We show an unfolded single-directional recurrent network in Figure 3, where each node at a single time-step represents a layer of network units. The network output at step t_i depends on both the current input at step t_i and the hidden state of t_{i-1}. The same weights (w1, w2, w3) are reused at every time-step.

LSTM. Unfortunately, with standard RNN architectures, the range of context that can be accessed is quite limited due to the vanishing gradient problem [14]. Long short-term memory (LSTM) [12] could address this problem by introducing a memory block which has the ability to preserve the state over long period of time. An LSTM network is similar to a standard RNN, except that the summation units in the hidden layer are replaced by memory blocks. Each block contains one or more self-connected memory cells and three multiplicative units (the input, output and forget gates). The three gates collect activation from inside and outside the block and control the activation of the cell via multiplications. The input and output gates multiply the input and output of the cell while the forget gate multiplies the cells previous state. The only output from the block to the rest of the network emanates from the output gate multiplication.

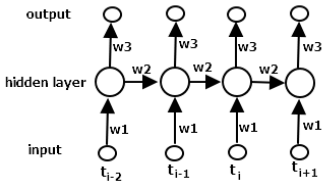


Fig. 3. An unfolded single-directional recurrent network

BLSTM. LSTM network processes the input sequence from past to future while Bidirectional LSTM [15], consisting of 2 separated LSTM layers, models the sequence from two opposite directions (past to future and future to past) in parallel. Both of 2 LSTM layers are connected to the same output layer. With this setup, complete long-term past and future context is available at each time step for the output layer.

Deep BLSTM. DBLSTM [16] can be created by stacking multiple BLSTM layers on top of each other in order to get higher level representation of the input data. The outputs of 2 opposite hidden layer at one level are concatenated and used as the input to the next level.

Non-chain-structured LSTM. A limitation of the network topology described above is that they only allow for sequential information propagation. Multidimensional LSTM [11] could memorize the information from n dimensions by introducing n forget gates in one memory block. In [8], the basic LSTM architecture was extend to tree structures, the Child-sum Tree-LSTM and the N-ary Tree-LSTM, allowing for richer network topology where each unit is able to incorporate information from multiple child units. In parallel to the work in [8], [9] explores the similar idea. The DAG-structured LSTM was proposed for semantic compositionality [10].

III. TREE-BASED BLSTM

This section will be focused on Tree-based BLSTM. Different with the tree structures depicted in [8], [9], we devote it to the kind of structures presented in Figure 4 where most nodes have only one next node. In fact, this kind of structure could be regarded as several chains with shared or overlapped segments. Traditional BLSTM process a sequence both from left to right

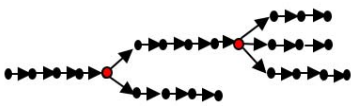


Fig. 4. A tree based structure for chains.

and from right to left in order to access information coming from two directions. In our case, the tree will be processed from root to leaves and from leaves to root.

From root to leaves. There are 2 special nodes (red) having more than one next node in Figure 4. We name them *Mul-next node*. The hidden states of *Mul-next node* will be propagated to its next nodes equally. The forward propagation of a *Mul-next nodes* is the same as for a chain LSTM node; with regard to the error propagation, the errors coming from all the next nodes will be summed up and propagated to *Mul-next node*.

From leaves to root. Suppose all the arrows in Figure 4 are reversed, the 2 red nodes are still special cases because they have more than one previous node. We call them *Mul-previous nodes*. The information from all the previous nodes will be summed up and propagated to the *Mul-previous node*; the error propagation is processed like for a typical LSTM node.

We give the specific formulas below regarding to the forward propagation of *Mul-previous node* and the error back-propagation of *Mul-next node*. The same notations as in [11] are used here. The network input to unit i at node n is denoted a_i^n and the activation of unit i at node n is b_i^n. w_{ij} is the weight of the connection from unit i to unit j. Considering a network with I input units, K output units and H hidden units, let the subscripts ς, ϕ, ω referring to the input, forget and output gate. The subscript c refers to one of the C cells. Thus, the peep-hole weights from cell c to the input, forget, output gates can be denoted as $w_{c\varsigma}$, $w_{c\phi}$, $w_{c\omega}$. s_c^n is the state of cell c at node n. f is the activation function of the gates, and g and h are respectively the cell input and output activation functions. L is the loss function used for training.

We only give the equations for a single memory block. For multiple blocks the calculations are simply repeated for each block. Let $\Pr(n)$ denote the set of previous nodes of node n and $\mathrm{Ne}(n)$ denote the set of next nodes.

The forward propagation of *Mul-previous node*

$$b_\varsigma^n = f(\sum_{i=1}^{I} w_{i\varsigma} x_i^n + \sum_{h=1}^{H} w_{h\varsigma} \sum_{p=1}^{|\Pr(n)|} b_h^p + \sum_{c=1}^{C} w_{c\varsigma} \sum_{p=1}^{|\Pr(n)|} s_c^p) \quad (1)$$

$$b_\phi^n = f(\sum_{i=1}^{I} w_{i\phi} x_i^n + \sum_{h=1}^{H} w_{h\phi} \sum_{p=1}^{|\Pr(n)|} b_h^p + \sum_{c=1}^{C} w_{c\phi} \sum_{p=1}^{|\Pr(n)|} s_c^p) \quad (2)$$

$$s_c^n = b_\phi^n \sum_{p=1}^{|\Pr(n)|} s_c^p + b_\varsigma^n g(\sum_{i=1}^{I} w_{ic} x_i^n + \sum_{h=1}^{H} w_{hc} \sum_{p=1}^{|\Pr(n)|} b_h^p) \quad (3)$$

$$b_\omega^n = f(\sum_{i=1}^{I} w_{i\omega} x_i^n + \sum_{h=1}^{H} w_{h\omega} \sum_{p=1}^{|\Pr(n)|} b_h^p + \sum_{c=1}^{C} w_{c\omega} s_c^n) \quad (4)$$

$$b_c^n = b_\omega^n h(s_c^n) \quad (5)$$

The error back-propagation of *Mul-next node*

We define

$$\epsilon_c^n = \frac{\partial L}{\partial b_c^n} \qquad \epsilon_s^n = \frac{\partial L}{\partial s_c^n} \qquad \delta_i^n = \frac{\partial L}{\partial a_i^n} \quad (6)$$

Then

$$\epsilon_c^n = \sum_{k=1}^{K} w_{ck} \delta_k^n + \sum_{g=1}^{4H} w_{cg} \sum_{e}^{|\mathrm{Ne}(n)|} \delta_g^e \quad (7)$$

$$\delta_w^n = f'(a_w^n) \sum_{c=1}^{C} h(s_c^n) \epsilon_c^n \quad (8)$$

$$\begin{aligned} \epsilon_s^n = {} & b_w^n h'(s_c^n) \epsilon_c^n + \sum_{e=1}^{|\mathrm{Ne}(n)|} b_\phi^e \sum_{e=1}^{|\mathrm{Ne}(n)|} \epsilon_s^e \\ & + w_{c\varsigma} \sum_{e=1}^{|\mathrm{Ne}(n)|} \delta_\varsigma^e + w_{c\phi} \sum_{e=1}^{|\mathrm{Ne}(n)|} \delta_\phi^e + w_{c\omega} \delta_\omega^n \end{aligned} \quad (9)$$

$$\delta_c^n = b_\varsigma^n g'(a_c^n) \epsilon_s^n \quad (10)$$

$$\delta_\phi^n = f'(a_\phi^n) \sum_{c=1}^{C} \sum_{p=1}^{|\Pr(n)|} s_c^p \epsilon_s^n \quad (11)$$

$$\delta_\varsigma^n = f'(a_\varsigma^n) \sum_{c=1}^{C} g(a_c^n) \epsilon_s^n \quad (12)$$

IV. Our Tree-based BLSTM Recognition System

The input data is available as a sequence of strokes S from which we would like to obtain the final LG graph describing unambiguously the ME. Let $S = (s_0, ..., s_{n-1})$, where we assume s_i has been written before s_j for $i < j$.

A. Derivation of an intermediate graph G

In a first step, we will derive an intermediate graph G where each node is a stroke and edges are added according to temporal or spatial relationships between strokes. Some definitions are introduced below for a better understanding of the derivation process.

Definition 2.1 A stroke s_i is considered visible from stroke s_j if the straight line between the centers of their bounding boxes does not cross the bounding box of any other stroke s_k.

Definition 2.2 We define five positional relations between two strokes s_i and s_j. The positional relation between them is simplified as the positional relation between the centers of their bounding boxes.

As shown in Figure 5, five regions ($R1$, $R2$, $R3$, $R4$ and $R5$) are defined for stroke s_i. If the center of bounding box of s_j is in one of these five regions, for example $R1$ region, we can say s_j is in the $R1$ direction of s_i. A wider searching range is defined for both $R3$ and $R4$ directions. That is because in some expressions like $\frac{a+b+c}{d+e+f}$, a larger searching range means

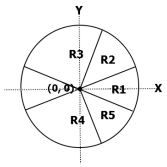
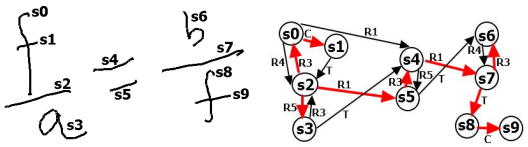


Fig. 5. Five directions for a stroke s_i. Point (0, 0) is the center of bounding box of s_i. The angle range of $R1$ region is $[-\frac{\pi}{8}, \frac{\pi}{8}]$; $R2 : [\frac{\pi}{8}, \frac{3*\pi}{8}]$; $R3 : [\frac{3*\pi}{8}, \frac{7*\pi}{8}]$; $R4 : [-\frac{7*\pi}{8}, -\frac{3*\pi}{8}]$; $R5 : [-\frac{3*\pi}{8}, -\frac{\pi}{8}]$.



(a) (b)

Fig. 6. (a) $\frac{f}{a} = \frac{b}{f}$ is written with 10 strokes; (b) the derived graph G, the red part is one of the possible trees with $s2$ as the root. $C : Crossing$, $T : Time$.

more possibilities to catch the $Above$ relationship from '$-$' to 'a' and the $Below$ relationship from '$-$' to 'd'.

Definition 2.3 Let G be a directed graph in which each node corresponds to a stroke and edges are added according to the following criteria in succession.

We defined for each stroke s_i (i from 0 to n-2):

- the set of crossing strokes $S_{\text{cro}(i)} = \{s_{\text{cro}1}, s_{\text{cro}2}, ...\}$ from $\{s_{i+1}, ..., s_{n-1}\}$.

For stroke s_i (i from 0 to n-1):

- the set $S_{\text{vis}(i)}$ of the visible rightmost strokes in five directions respectively.

Edges from s_i to the $S_{\text{cro}(i)} \bigcup S_{\text{vis}(i)}$will be added to G. Then, we check if the edge from s_i to s_{i+1} (i from 0 to n-2) exists in G. If not, this edge is added to G to ensure that the path covering the sequence of strokes in the time order is included in G. Each edge is tagged depending on the specific criterion we used to find it before. Consequently, we have at most 7 types of edges ($Crossing, R1, R2, R3, R4, R5$ and $Time$) in the graph. For those edges from s_i to the $S_{\text{cro}(i)} \cap S_{\text{vis}(i)}$, the type $Crossing$ is assigned.

B. Derivation of trees from G

Figure 6 illustrates the ME $\frac{f}{a} = \frac{b}{f}$ written with 10 strokes and the corresponding G. We would like to label nodes and edges of G correctly to build a SLG finally. The solution adopted in this study is deriving trees from G, then recognizing the trees using the tree-based BLSTM.

There exists different strategies to derive trees from G. First of all, a start node should be selected. We take the left most stroke as the starter. For the example mentioned in the last paragraph, stroke $s2$ is the starter. From the starting node, we traverse the graph with the Depth-First Search algorithm. Each node should be visited only once. When there are more than one edge outputting from one node, the visiting order will follow $(Crossing, R1, R4, R3, R2, R5, Time)$. With this strategy, a tree is derived to which we give the name *Tree-Left*. In Figure 6b, *Tree-Left* is depicted in red with the root in $s2$. Note that in this case, all the nodes are accessible from the start node $s2$. However, as G is a directed graph, some nodes are not reachable from the starter in some cases. Therefore, one tree might not model all the strokes and the relationships between them for some expressions. Now, suppose that instead of the left-most stroke we use stroke $s0$ as the starting node and keep the same strategy to derive the tree. This new tree is called *Tree-0*. Finally, if $s0$ is taken as the starting point and time order is considered first, a special tree *Tree-Time* is obtained which is a chain structure in fact. *Tree-Time* is defined by $s0 \rightarrow s1 \rightarrow s2 \rightarrow s3 \ldots \rightarrow s9$ for the ME in Fig. 6.

C. Feed the inputs of the Tree-based BLSTM

In section IV-B, we derived trees from the intermediate graph. Nodes of the tree represent visible strokes and edges denote the relationships between pairs of strokes. We would like to label each node and edge correctly aiming to build a complete SLG finally.

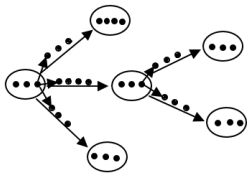


Fig. 7. A re-sampled tree. The small arrows between points provide the directions of information flows. With regard to the sequence of points inside one node or edge, most of small arrows are omitted.

The next step is to go from the previous trees defined at the stroke level down to a tree at the point level, points being the raw information that are recorded along the pen trajectory in the online signal. In the considered trees, nodes, which represent strokes, are re-sampled with a fixed spatial step, and the same holds for edges by considering the straight lines in the air between the last point and the first point of a pair of strokes that are connected in the tree. This is illustrated in Fig. 7, where the re-sampled points are displayed inside the nodes and above the edges. Since this tree will be processed by the BLSTM network, we need for the training stage to assign it a corresponding ground-truth. We derive it from the SLG by using the corresponding symbol label of the strokes (nodes) for the on-paper points and the corresponding symbol or relationship label for the in-air points (edges) when this edge exists in the SLG. When an edge of the tree does not exist in the SLG, the label $NoRelation$ noted '_' will be used. In this way, an edge in the graph which was originally denoted with a C, Ri $(i = 1...5)$ or T relation will be assigned one of the 7 labels: $(Right, Above, Below, Inside, Superscript, Subscript, \_)$ or a symbol label when the two strokes are belonging to the same symbol.

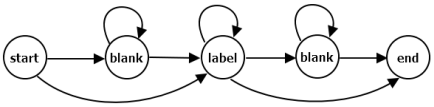


Fig. 8. The possible labels of points in one short sequence.

D. Training process

The system processes each node or edge separately but following the order with which the correct propagation of activation or errors could be ensured. As a matter of fact, each node or edge is a short sequence. It is known that BLSTM and CTC stage have better performance when a "blank" label is introduced during training [17], so that decision can be made only at some point in the input sequence. Globally, the data structure we are dealing with is a tree; locally, it consists of several short sequences. Inside each short sequence, or we can say each node or edge, the CTC loss function could be computed. Since each node or edge belongs to one class, the possible labels of points are shown in Fig. 8. We refer the readers to [18] for a detailed description.

E. Recognition process

As mentioned, the system treats each node or edge as a short sequence. A simple decoding method is adopted here. We choose for each node or edge the label which has the highest cumulative probability over the short sequence. Suppose that p_{ij} is the probability of outputting the i label at the j point. The probability of outputting the i label can be computed as $P_i = \sum_{j=1}^{s} p_{ij}$, where s is the number of points in a short sequence. The label with the highest probability is assigned to this short sequence.

F. Post process

Several trees related to one ME will be merged to build a LG after labeling. Each node or edge belongs at least to one tree, but possibly to several trees. Hence, several recognition results can be available for a single node or edge. We take an intuitive and simple way to deal with the problem of multiple results, choosing the one with the highest probability.

Afterwards, some edges can be added automatically to complete a valid LG. We first look for the symbols using connected component analysis: a connected component where nodes and edges have the same label is a symbol. With regards to the relationship between two symbols, we choose the label having the maximum accumulative probability among the edges between two symbols. Then, according to the rule that all strokes in a symbol have the same input and output edges and that double-direction edges represent the segments, some missing edges can be completed automatically.

V. Experiments

Features. A stroke is a sequence of points sampled from the trajectory of a writing tool between a pen-down and a pen-up at a fixed interval of time. Then an additional re-sampling is performed with a fixed spatial step to get rid of the writing speed. The number of re-sampling points depends on the size of expression. For each node or edge, we re-sample with $10 \times l/d$ points. Here, l refers to the length of a visible stroke or a straight line connecting 2 strokes and d refers to the average diagonal of the bounding boxes of all the strokes in an expression.

Subsequently, we compute five features per point, which are quite close to the state of art [7], [3]. For every point $p(x, y)$ we obtained 5 features [sinθ, cosθ, sinϕ, cosϕ, PenUD]. The detailed description can be seen in [18].

Data set. The complete data set from CROHME 2014 is used, 8834 expressions for training and 982 expressions for test. We extract 10% of the 8834 expressions of the training set as a validation set. For each expression, *Tree-Time*, *Tree-Left* and *Tree-0* were derived to train three classifiers separately.

Setup. We constructed the tree-based BLSTM recognition system with the RNNLIB library[1]. Two types of configurations are included in this paper: *Network (i)* and *Network (ii)*. The first one consists of one bidirectional hidden level (two opposite LSTM layers of 100 cells). This configuration has obtained good results in both handwritten text recognition [6] and handwritten math symbol classification [7]. *Network (ii)* is a deep structure with two bidirectional hidden levels, each containing two opposite LSTM layers of 100 cells. The setup about the input layer and output layer remains the same. The size of the input layer is 5 (5 features); the size of the output layer is 109 (101 symbol classes + 6 relationships + $NoRelation$ + $blank$).

Evaluation. With the Label Graph Evaluation library (LgEval) [19], the recognition results can be evaluated on symbol level and on expression level. We introduce several evaluation criteria: symbol segmentation (Segments), refers to a symbol that is correctly segmented whatever the label is; symbol segmentation and recognition (Seg+Class), refers to a symbol that is segmented and classified correctly; spatial relationship classification (Tree Rels.), a correct spatial relationship between two symbols requires that both symbols are correctly segmented and with the correct relationship label.

Discussion. The evaluation results on symbol level and global expression level are presented in Table I and II respectively. We give both the individual tree recognition results and the merging results in each table. *Tree-Time* covers all the strokes of the input expression but can miss some relational edges between strokes; *Tree-Left* and *Tree-0* could catch some additional edges which are not covered by *Tree-Time*. The experiment results also verified this tendency. Compared to (i, *Tree-Time*), the symbol segmentation and classification results of (i, *Merge3*) are improved in a moderate amount while the recall rate of relationship classification is greatly improved (about 10%) with the precision rate remaining almost the same. The different recognition results of network (ii) are systematically increased when compared to (i) as the deep structure could get higher level representations of the input

[1]Graves A. RNNLIB: A recurrent neural network library for sequence learning problems. http://sourceforge.net/projects/rnnl/.

TABLE I
THE SYMBOL LEVEL EVALUATION RESULTS ON CROHME 2014 TEST SET, INCLUDING THE EXPERIMENT RESULTS IN THIS WORK AND CROHME 2014 PARTICIPANT RESULTS (TOP 4 BY RECALL OF SEGMENTS).

Network, model	Segments (%)		Seg + Class (%)		Tree Rels. (%)	
	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.
i, *Tree-Time*	92.85	86.00	84.41	78.18	58.04	76.99
i, *Tree-Left*	86.84	77.93	75.98	68.18	52.24	65.68
i, *Tree-0*	89.22	79.56	77.98	69.54	59.63	62.06
i, *Merge3*	**93.39**	**87.46**	**85.95**	**80.49**	**69.52**	**74.92**
ii, *Tree-Time*	95.10	90.47	87.53	83.27	65.06	83.18
ii, *Tree-Left*	89.34	82.17	78.90	72.57	56.13	73.21
ii, *Tree-0*	90.15	81.39	80.09	72.31	55.53	68.05
ii, *Merge3*	**95.59**	**91.46**	**89.02**	**85.17**	**74.29**	**81.33**
system	CROHME 2014 participant results (Top 4)					
III	98.42	98.13	93.91	93.63	94.26	94.01
I	93.31	90.72	86.59	84.18	84.23	81.96
VII	89.43	86.13	76.53	73.71	71.77	71.65
V	88.23	84.20	78.45	74.87	61.38	72.70

TABLE II
THE EXPRESSION LEVEL EVALUATION RESULTS ON CROHME 2014 TEST SET, INCLUDING THE EXPERIMENT RESULTS IN THIS WORK AND CROHME 2014 PARTICIPANT RESULTS (TOP 4)

Network, model	correct (%)	$\leq$ 1 error	$\leq$ 2 errors	$\leq$ 3 errors
i, *Tree-Time*	11.20	18.43	26.07	31.16
i, *Tree-Left*	7.33	13.95	18.64	22.51
i, *Tree-0*	8.45	15.99	21.49	26.27
i, *Merge3*	**17.11**	**25.46**	**32.28**	**38.19**
ii, *Tree-Time*	16.09	25.46	32.28	37.27
ii, *Tree-Left*	9.16	17.11	21.59	25.66
ii, *Tree-0*	9.67	15.99	21.69	27.49
ii, *Merge3*	**21.59**	**30.96**	**37.88**	**43.99**
system	CROHME 2014 participant results (Top 4)			
III	62.68	72.31	75.15	76.88
I	37.22	44.22	47.26	50.20
VII	26.06	33.87	38.54	39.96
VI	25.66	33.16	35.90	37.32

data. With regard to the symbol classification and recognition rates, our system (ii) performs better than the second-ranked system in CROHME 2014. For relationship classification rate, our system (ii) reaches the level between the second-ranked and the third-ranked systems in CROHME 2014. And it should be improved continually as we consider more trees in future work. The global expression recognition rate is 21.59%. When we compute the recognition rate with $\leq$ 3 errors, our result is 43.99%, higher than the third-ranked system (39.96%). The top ranked system is from a company and they use a much larger training data set which is not available to the public. Furthermore, as we know, all the top 4 systems in the CROHME 2014 competition are grammar driven solutions which need a large amount of manual work and a high computational complexity. Considering that there is no grammar constraint included in our system, the results are quite promising.

VI. CONCLUSION

We proposed a tree-based BLSTM system for online ME recognition. One major difference with the traditional approaches is that there is no explicit segmentation, recognition and layout extraction steps but a unique trainable system that produces directly a SLG describing a ME. The current system, without any grammar, achieves competitive results on symbol level. With regard to the recognition rate on global expression level, there is still room for improvement. In future, we will consider more trees with the aim to catch more relationships between strokes. Another direction is to extend the tree-based BLSTM to DAG structure, avoiding the problem of merging the results from several trees.

ACKNOWLEDGMENT

We would like to thank the China Scholarship Council for supporting PhD studentship at Nantes university.

REFERENCES

[1] H. Mouchère, R. Zanibbi, U. Garain, C. Viard-Gaudin. *Advancing the state of the art for handwritten math recognition: the CROHME competitions, 2011-2014.* IJDAR, 19(2): 173-189, 2016.

[2] R. Zanibbi, D. Blostein. *Recognition and retrieval of mathematical expressions.* IJDAR, 15(4): 331-357, 2012.

[3] A. M. Awal, H. Mouchère, C. Viard-Gaudin. *A global learning approach for an online handwritten mathematical expression recognition system.* PRL, 35: 68-77, 2014.

[4] F. Álvaro, J. A. Sánchez, J. M. Benedí. *An integrated grammar-based approach for mathematical expression recognition.* PR, 35: 135-147, 2016.

[5] T. Zhang, H. Mouchère, C. Viard-Gaudin. *Online Handwritten Mathematical Expressions Recognition by Merging Multiple 1D Interpretations.* ICFHR, 2016.

[6] A. Graves, M. Liwicki, S. Fernández, et al. *A novel connectionist system for unconstrained handwriting recognition.* IEEE Transactions on PAMI, 31(5): 855-868, 2009.

[7] F. Álvaro, J. A. Sánchez, J. M. Benedí. *Classification of on-line mathematical symbols with hybrid features and recurrent neural networks.* 2013 12th ICDAR. IEEE, 1012-1016, 2013.

[8] K. S. Tai, R. Socher, and C. D. Manning. *Improved semantic representations from tree-structured long short-term memory networks.* arXiv preprint arXiv: 1503.00075, 2015.

[9] X. Zhu, P. Sobhani, and H. Guo. *Long Short-Term Memory Over Recursive Structures.* ICML, 1604-1612, 2015.

[10] X. Zhu, P. Sobhani, and H. Guo. *Dag-structured long short-term memory for semantic compositionality.* Proceedings of NAACL-HLT, 917-926, 2016.

[11] A. Graves. *Supervised sequence labelling with recurrent neural networks.* Heidelberg: Springer, 2012.

[12] S. Hochreiter, J. Schmidhuber. *Long short-term memory.* Neural computation, 9(8): 1735-1780, 1997.

[13] R. Zanibbi, H. Mouchère, C. Viard-Gaudin. *Evaluating structural pattern recognition for handwritten math via primitive label graphs.* Document Recognition and Retrieval XX, Feb 2013, Burlingame, United States. 8658, pp.865817-865817-11, 2013.

[14] S. Hochreiter, Y. Bengio, P. Frasconi, et al. *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.* 2001.

[15] A. Graves, S. Fernández, J. Schmidhuber. *Bidirectional LSTM networks for improved phoneme classification and recognition.* ICANN, Springer Berlin Heidelberg, 2005: 799-804.

[16] A. Graves, N. Jaitly, A. Mohamed. *Hybrid speech recognition with deep bidirectional LSTM.* Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on. IEEE, 2013: 273-278.

[17] T. Bluche, H. Ney, J. Louradour, et al. *Framewise and ctc training of neural networks for handwriting recognition.* ICDAR, 2015: 81-85.

[18] T. Zhang, H. Mouchère, C. Viard-Gaudin. *Using BLSTM for interpretation of 2-D languages.* Document numérique, 2016, DOI : 10.3166/DN.19.2-3.135-157.

[19] H. Mouchère, C. Viard-Gaudin, R. Zanibbi, et al. *ICFHR 2014 Competition on Recognition of On-line Handwritten Mathematical Expressions (CROHME 2014).* ICFHR, 2014.