Graph-to-Graph: Towards Accurate and Interpretable Online Handwritten Mathematical Expression Recognition

Jin-Wen Wu[1,2*] **, Fei Yin**[1]**, Yan-Ming Zhang**[1]**, Xu-Yao Zhang**[1,2]**, Cheng-Lin Liu**[1,2,3]
[1] National Laboratory of Pattern Recognition, Institute of Automation of Chinese Academy of Sciences
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences
[3] CAS Center for Excellence of Brain Science and Intelligence Technology
{jinwen.wu, fyin, ymzhang, xyz, liucl}@nlpr.ia.ac.cn

Abstract

Recent handwritten mathematical expression recognition (HMER) approaches treat the problem as an image-to-markup generation task where the handwritten formula is translated into a sequence (e.g. LaTeX). The encoder-decoder framework is widely used to solve this image-to-sequence problem. However, (i) for structured mathematical formula, the hierarchical structure neither in the formula nor in the markup has been explored adequately. In addition, (ii) existing image-to-markup methods could not explicitly segment mathematical symbols in the formula corresponding to each target markup token. In this paper, we address the above issues by formulating the HMER as a *graph-to-graph* (G2G) *learning* problem. Graph is more flexible and general for structure representation and learning compared with image or sequence. At the core of our method lies the embedding of input formula and output markup into graphs on primitives, with Graph Neural Networks (GNN) to explore the structural information, and a novel sub-graph attention mechanism to match primitives in the input and output graphs. We conduct extensive experiments on CROHME datasets to demonstrate the benefits of the proposed G2G model. Our method yields significant improvements over previous SOTA image-to-markup systems. Moreover, it explicitly resolves the symbol segmentation problem while still being trained end-to-end, making the whole system much more accurate and interpretable.

Introduction

Mathematical notation has essential applications in many fields, such as education, office automation, and conference systems. Recognizing mathematical formulas requires the ability to analyze both their *mathematical semantics* and complex *hierarchical structures*. As early as 1960s, there has been research interest (Anderson 1967) in converting mathematical images into structured language or *markup* (e.g. LaTeX, MathML and Symbol Layout Tree (SLT, Zanibbi and Blostein 2012)) that identifies symbol descriptions and associated structures. See an example in Fig. 1.

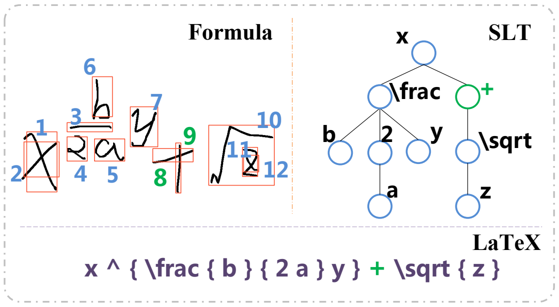

Figure 1: Handwritten formula with its SLT and LaTeX format markups. Primitives in the formula and markup have hierarchical relations. Besides, each primitive in the formula has specific representation in the markup. For example, the 8th and 9th strokes in the input formula correspond to the node "+" in SLT and the token "+" in LaTeX.

Recently, inspired by the success of sequence-to-sequence learning in different applications such as speech recognition (Chorowski et al. 2015), machine translation (Bahdanau, Cho, and Bengio 2014), and image captioning (Xu et al. 2015; Li et al. 2017), the encoder-decoder framework has also been proposed for HMER. Essentially, these approaches treat HMER as a special case of general image-to-sequence problem, that is, image-to-markup generation (Deng et al. 2017). Given a source formula input $\mathbf{x}$ (e.g. image or handwriting trajectory) and its target representation $\mathbf{y}$, the image-to-markup system models $\mathbf{x} \rightarrow \mathbf{y}$ from the training data. Generally, $\mathbf{y} = (y_1, \cdots, y_T)$ is a LaTeX string with T tokens. In fact, the LaTeX string is a tree with additional formatting annotations and can be obtained by depth-first traversal of the SLT. However, this kind of information is actually ignored in image-to-sequence approaches.

Despite recent progress in image-to-markup based HMER, these approaches could not achieve accurate and satisfactory results. The defects of these methods are mainly in the following aspects: (i) the input representation of image or handwriting trajectory makes the encoder unable to effectively explore the spatial and temporal structures among input primitives. (ii) Although formulas have linear forms markup, i.e. LaTeX string, the underlying structure of

*Corresponding author.

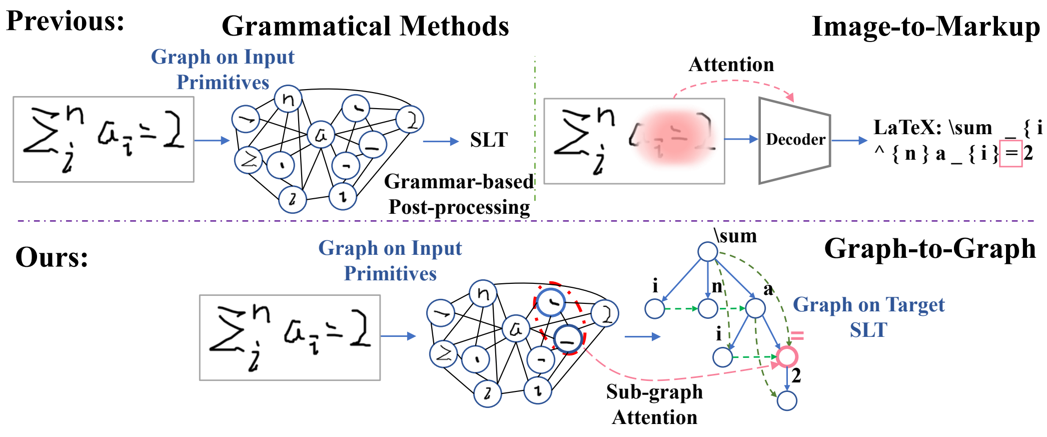

Figure 2: Brief description of different HMER approaches. We construct the source graph on input primitives (i.e. handwriting strokes) and the target graph based on SLT format markup. Compared with existing HMER systems, our G2G model explores the hierarchical relationship among primitives both in the formula and the markup. It explicitly segments symbols in the input formula while still being trained end-to-end.

formulas is known to be hierarchical. Sequential decoder could not explicitly leverage the hierarchical structure embedded in the markup. (iii) Moreover, during generation, the decoder "looks" at the relevant regions of each token $y_t, t = \{1, \cdots, T\}$ with an implicit segmentation function $Attn(\mathbf{x}, t)$, i.e. the vanilla attention mechanism, and is unable to explicitly segment the primitives $x_n \in \mathbf{x} = (x_1, \cdots, x_N), n = \{1, \cdots, N\}$ corresponding to y_t. For example, see the process of identifying "=" in Fig. 2.

In addition to image-to-markup methods, grammatical methods (Julca-Aguilar et al. 2020) were also proposed for HMER. Although such methods can explicitly parse the primitives corresponding to the target token, they require complicated manual work to design grammars, and thus, are less efficient compared with data driven learning.

To solve the problems described above, in this paper, we formulate HMER as a *graph-to-graph learning* problem $G_{\mathbf{x}} \rightarrow G_{\mathbf{y}}$. Specifically, (i) $G_{\mathbf{x}} = (V_{\mathbf{x}}, E_{\mathbf{x}})$ is a directed graph on the handwritten strokes of input formula $\mathbf{x}$, where $V_{\mathbf{x}}$ is the node (or vertex) set corresponding to the input strokes, and $E_{\mathbf{x}}$ is the edge set that records the spatial and temporal relationship between strokes. (ii) $G_{\mathbf{y}} = (V_{\mathbf{y}}, E_{\mathbf{y}})$ is an directed graph constructed on the SLT of $\mathbf{x}$. Representing the markup as a graph allows explicit use of the hierarchical information contained therein. (iii) Then, association between these two graphs is learned via a novel sub-graph attention mechanism. By consolidating the idea of modeling HMER as G2G learning, we present a novel GNN-GNN architecture to model the primitive representations and relationships in the source and target graphs.

The main contributions of our work are highlighted as follows:

- We formulate HMER as a G2G learning problem. GNN-based encoder and decoder are exploited to learn the structural information of the input formula and output markup from training data.
- Our G2G model is an end-to-end approach for explicit symbol segmentation, symbol identification, and hierarchical structure analysis, and so, is more accurate and interpretable.
- The proposed method establishes new SOTA results on the benchmark datasets CROHME 2014 and 2016, pushing the expression recognition rate (ExpRate) from 50.41% to 54.46%, and 49.61% to 52.05%, respectively.

The rest of this paper is organized as follows. Section 2 presents a brief review to HMER approaches. Section 3 details the proposed G2G model. Section 4 presents extensive ablation experiments and comparison experiments to evaluate our method. Finally, concluding remarks are drawn in Section 5.

Related Work

HMER requires to convert the input formula (in the formats of either image or handwriting trajectory) into a structured representation, which contains both mathematical symbols and their placement on the writing line. In the following, we review the main HMER approaches.

Grammatical Methods

Recognizing a formula usually involves three steps: symbol segmentation, symbol recognition, and structural analysis. Most grammatical methods apply modifications of context-free string grammars to solve these tasks (Alvaro, Sánchez, and Benedí 2014, 2016). However, such grammar limits the inclusion of at most two elements on the right side of the rule. To overcome this limitation, other manually designed string grammars, such as fuzzy relational grammar (MacLean and Labahn 2013), have been proposed.

Compared with string grammar, graph grammar can better represent the hierarchical structure of formula. Several graph grammar based methods have been investigated for recognizing formula as graph (Lavirotte 1997; Julca-Aguilar et al. 2020). There are also some approaches (Mahdavi et al.

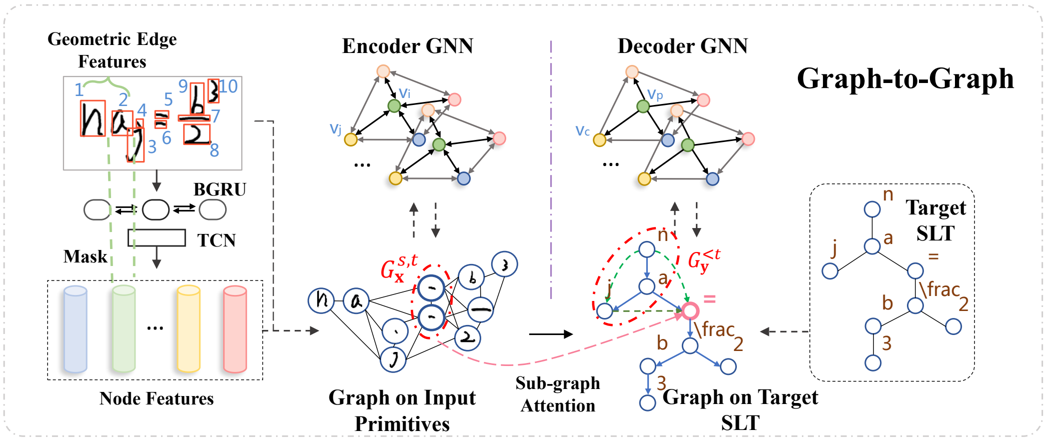

Figure 3: An overview of our G2G model for HMER. The source graph (left) is built on the input primitives (i.e. online stokes) and target graph is construct on the SLT. Each node in the target graph are generated conditioned on the sub-graphs both in the input and output. Association between the generated node, such as node "=" in the target graph, and the source sub-graph are learned with a sub-graph attention. By learning from G2G, our model enables accurate, interpretable and end-to-end HMER.

2019; Mahdavi, Sun, and Zanibbi 2020) directly generate a second graph on symbols by merging nodes of the source graph on primitives and identify spatial relationships from edges without using graph grammars.

Image-to-Markup

To make the training data-driven and end-to-end, image-to-markup method based on encoder-decoder framework has been proposed (Deng et al. 2017). Such methods use a CNN or RNN based encoder to embed the input formula, and an attention based sequential decoder to generate the target markup. Specially, an image-to-markup approach, named PAL (Wu et al. 2018), trains the recognizer with paired adversarial learning to overcome the writing-style variation faced by handwritten formulas. Its improved version PAL-v2 (Wu et al. 2020) utilizes a pre-aware unit to enhance the accuracy of attention. A dual loss attention based method (Le 2020) shares similar idea with PAL in learning semantic invariant features with the guide of printed templates.

Different encoders, such as encoders based on VGG (Zhang et al. 2017) and multi-branch DenseNet (Zhang, Du, and Dai 2018) have been investigate to improve performance. Some works use bidirectional RNNs to directly encode online formula strokes (Le and Nakagawa 2017; Zhang, Du, and Dai 2018). There are also image-to-markup approaches which apply data augmentation (Le and Nakagawa 2017) and multi-modal ensemble (Wang et al. 2019) methods to investigate the impact of training data. A parallel work (Zhang et al. 2020) explicitly considers the tree-structured markup with a parent-child decoder.

Summary

As described above, grammatical methods could not automatically learn from the markup corpus. On the other hand, image-to-markup methods do not effectively explore the hierarchical structure embedded both in the formula and the markup. In addition, the latter could not explicitly segment symbols in the input corresponding to output token.

Recently, neural network structure operating on graphs has received a surge of attention (Yao et al. 2018; Liu et al. 2019). Compared with image or sequence representations for formula, graph representations are more natural and general. Modeling HMER in the form of graphs makes the recognition process more accurate and interpretable.

Method

Given an input formula $\mathbf{x}$ (e.g. online handwriting strokes or the image rendered by them) and its LaTeX format markup $\mathbf{y} = (y_1, \cdots, y_T)$, the image-to-markup system infers $\mathbf{y}$ by modeling:

$$p(\mathbf{y}|\mathbf{x}) = \prod_t p(y_t|y_{<t}; Attn(\mathbf{x}, t)), \tag{1}$$

where $y_{<t}$ denotes previous $t-1$ tokens before y_t. Intuitively, mathematical formula is a kind of structured data. However, (i) hierarchical structures in both $\mathbf{x}$ and $\mathbf{y}$ are not explored adequately, and (ii) the vanilla attention mechanism $Attn(\cdot)$ could not explicitly segment primitives corresponding to y_t. To this end, we propose to treat the structured input and output as graphs. Fig. 3 illustrates our method. We redefine the Equation 1 as:

$$p(G_{\mathbf{y}}|G_{\mathbf{x}}) = \prod_t p((v^t_{\mathbf{y}}, E^t_{\mathbf{y}})|G^{<t}_{\mathbf{y}}; G^{s,t}_{\mathbf{x}}), \tag{2}$$

where $v^t_{\mathbf{y}}$ denotes the generated node at step t, $E^t_{\mathbf{y}}$ is the generated directed edge set from the existing target graph $G^{<t}_{\mathbf{y}}$ to $v^t_{\mathbf{y}}$, and $G^{s,t}_{\mathbf{x}}$ indicates the source sub-graph corresponding to $v^t_{\mathbf{y}}$. We now present details of our G2G model.

Representations of Source and Target Graph

Graph on Input Primitives In this work, we use online strokes $\mathbf{x} = (x_1, \cdots, x_N)$ which make up the mathematical symbols as the primitives of the input graph $G_{\mathbf{X}} = (V_{\mathbf{X}}, E_{\mathbf{X}})$, where $V_{\mathbf{X}}$ is the node set corresponding to input strokes, and $E_{\mathbf{X}}$ is the edge set composed of spatial edges constructed with line-of-sight (LOS) algorithm (Hu and Zanibbi 2016) and bidirectional temporal edges between time neighbors. For the n-th stroke x_n, $n = \{1, \cdots, N\}$, with L_n track points, the corresponding node $v_{\mathbf{X}}^n$ is embedded by:

$$\mathbf{f}_0^n = F(\mathbf{x})\mathbf{k}_n / ||\mathbf{k}_n||_1, \tag{3}$$

where $\mathbf{k}_n = (k_1^n, \cdots, k_L^n) \in \mathbb{R}^L$ is a *binary mask*, $L = \sum_n L_n$ is the input trajectory length, $k_l^n = 1$ when $\sum_{i=1}^{(n-1)} L_i < l \leq \sum_{i=1}^{n} L_i$ and otherwise $k_l^n = 0$, and $F(\mathbf{x}) \in \mathbb{R}^{C \times L}$ is the feature sequence encoded by cascaded deep neural network blocks. Each block consists of a bidirectional GRU and a subsequent Temporal Convolutional Network (TCN, Gehring et al. 2017), activated by the Gated Linear Uint (GLU, Dauphin et al. 2017). Besides, for each directed edge $e_{\mathbf{X}}^{i,j}, i, j = \{1, \cdots, n\}$, between a pair of input primitives, we extract a geometric feature (Ye et al. 2019) and project the feature into $\mathbf{b}_0^{i,j} \in \mathbb{R}^C$.

Graph on Target SLT The SLT, $G_{SLT} = (V_{SLT}, E_{SLT})$, of a mathematical formula records both symbols and structural relationship between different symbols. Our goal is to explicitly use hierarchical information embedded in it.

To this end, we set $V_{\mathbf{Y}} = V_{SLT}$ and design three types of edges to represent the hierarchical structure, i.e. directed edges pointing to the current node from its grandparent (E_{gg}), parent (E_{pc}), and left brother (E_{bb}) in the SLT, respectively. For convenience, we index the nodes of STL in depth-first order. Moreover, to guarantee that the decoder can decide whether current node is the last child node of its parent, we modify the SLT by adding an end child node to each node as the rightmost child. The relationship between the end child node and its parent is represented as "_".

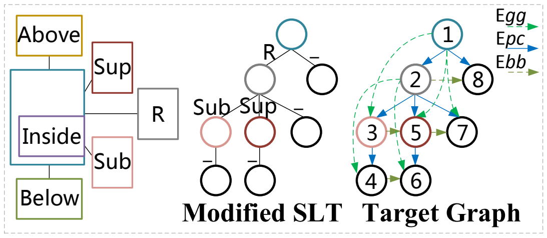

Figure 4: Modified SLT and the corresponding target graph. Spatial relationship between symbol in the blue box and symbol in other boxes (left) is recorded by the parent-child edge in SLT. Black circle (center and right) represents the added end child node.

Graph-to-Graph Learning

Encoder GNN The representation of the source graph $G_{\mathbf{X}}$ is modeled with a modified Graph Attention Network (GAT, Velickovic et al. 2018). For the q-th GAT layer, we first update the edge feature $\mathbf{b}_q^{i,j}$ by:

$$\mathbf{b}_q^{i,j} = \delta(\mathbf{W}_{b,q}[\mathbf{f}_{q-1}^i; \mathbf{b}_{q-1}^{i,j}; \mathbf{f}_{q-1}^j]), \tag{4}$$

where $\mathbf{W}_{b,q} \in \mathbb{R}^{C \times (2C+C)}$ is the weight matrix, $\delta(\cdot)$ is the LeakyRelu activation function, and ";" denotes the concatenation. Updating the edge embedding by combining the visual feature of the current node i and its neighborhood potentially enables the encoder to learn where to attend given the current node.

After that, the hidden features for every node are computed based on the self-attention mechanism:

$$\mathbf{f}_q^i = \delta(\sum_{j \in \mathcal{N}_i} a_{i,j} \mathbf{W}_{f,q} \mathbf{f}_{q-1}^j), \tag{5}$$

where $a_{i,j}$ is the attention coefficient, $\mathbf{W}_{f,q} \in \mathbb{R}^{C \times C}$ is a weight matrix, and $\mathcal{N}_i$ is the neighborhood of the current node i. The attention mechanism is designed as:

$$a_{i,j} = \frac{\exp(\beta r_{i,j})}{\sum_{k \in \mathcal{N}_i} \exp(\beta r_{i,k})}, \tag{6}$$

where β is a temperature set to 1 in this work, and $r_{i,j}$ is calculated as:

$$r_{i,j} = \delta(\mathbf{u}_{b,q}^T(\mathbf{W}_{b,q}^{'} \mathbf{b}_q^{j,i} + \mathbf{W}_{f,q}^{'} \mathbf{f}_{q-1}^i + \mathbf{W}_{f,q}^{'} \mathbf{f}_{q-1}^j)), \tag{7}$$

where $\mathbf{u}_{b,q} \in \mathbb{R}^C$ is a weight vector, $\mathbf{W}_{b,q}^{'}, \mathbf{W}_{f,q}^{'} \in \mathbb{R}^{C \times C}$ are learnable parameters. When set β as zero, the model degrades to the vanilla Graph Convolutional Network (GCN).

Decoder GNN Taking the inspiration from *sequence-to-sequence learning* (Gehring et al. 2017), we devise an attentional GNN-based decoder to learn the representation of target graph $G_{\mathbf{Y}}$ and the association between $G_{\mathbf{X}}$ and $G_{\mathbf{Y}}$. Each block of the decoder is equipped with a modified GCN layer and subsequent attention mechanism. Embedded node feature $\mathbf{z}_m^t \in \mathbb{R}^C, t = \{1, \cdots, |V_{\mathbf{Y}}| = T\}$ of the m-th decoder block is updated based on both the previous block output and related node features $\mathbf{f}^n \in \mathbb{R}^C, n = \{1, \cdots, |V_{\mathbf{X}}| = N\}$ of the source graph. Specially, $\mathbf{z}_0^t$ is the word embedding corresponding to target node t.

Formally, for each modified GCN layer, we leverage separate weight parameters for different types of edges to make the decoder sensitive to the hierarchical structure. Accordingly, the node output of the GCN layer in the m-th block is computed by:

$$\mathbf{h}_m^t = \rho(\sum_{j \in \mathcal{N}_t} \mathbf{W}_{d(j,t),m} \mathbf{z}_{m-1}^j). \tag{8}$$

where $d(j,t)$ dynamically selects the learnable weight matrix $\mathbf{W}_{d(j,t),m} \in \mathbb{R}^{2C' \times C}$ with regard to the type of each edge (i.e. $\mathbf{W}_{gg,m}$ for grandparent-to-grandchild, $\mathbf{W}_{pc,m}$ for parent-to-child, $\mathbf{W}_{bb,m}$ for brother-to-brother, and $\mathbf{W}_{cc,m}$ for current-to-current), and $\rho(\cdot)$ is the GLU activation function. Specially, the word embedding $\mathbf{z}_0^t$ of current node t is set to zero for the first GCN layer, since it is unavailable in the test.

Then, the target node embedding is computed by injecting the context vector $\mathbf{c}_m^t$ of source graph:

$$\mathbf{z}_m^t = \mathbf{W}_{z,m}(\mathbf{h}_m^t + \mathbf{c}_m^t), \tag{9}$$

where $\mathbf{W}_{z,m} \in \mathbb{R}^{C \times C'}$ is a weight matrix, and the context vector is computed with the attention mechanism:

$$\mathbf{c}_m^t = \sum_{n=1}^{N} \alpha_m^{t,n} \mathbf{W}_V \mathbf{f}^n. \tag{10}$$

$\mathbf{W}_V \in \mathbb{R}^{C' \times C}$ projects the source node feature as the value, and the attention coefficient $\alpha_m^{t,n}$ is obtained by:

$$\mathbf{q}_m^t = \mathbf{W}_{h,m}\mathbf{h}_m^t + \mathbf{z}_0^{bro(t)} + \mathbf{z}_0^{pa(t)}, \tag{11}$$

$$\alpha_m^{t,n} = \frac{\exp(\mathbf{q}_m^t \cdot (\mathbf{W}_K \mathbf{f}^n))}{\sum_{j=1}^{N} \exp((\mathbf{q}_m^t \cdot (\mathbf{W}_K \mathbf{f}^j)))}, \tag{12}$$

where $\mathbf{W}_{h,m} \in \mathbb{R}^{C \times C'}$ is a weight matrix, $\mathbf{W}_K \in \mathbb{R}^{C \times C}$ embed $\mathbf{f}^n$ as a key, and $bro(\cdot)$ and $pa(\cdot)$ are the functions selecting the left brother and parent of node t, respectively.

Encoder Supervision The training object of the proposed G2G model includes the representation learning of both $G_{\mathbf{x}}$ and $G_{\mathbf{y}}$. The output node embedding of the encoder GNN is taken for learning the category of the mathematical symbol to which the corresponding stroke belongs:

$$\hat{p}_{f,n} = \mathrm{softmax}(\mathbf{W}_{f,o}\mathbf{f}^n), \tag{13}$$

where $\mathbf{W}_{f,o}$ are the parameters of an Multi-Layer Perceptron (MLP). The classification loss of nodes is defined as:

$$\mathcal{L}_{nf} = -\sum_{n=1}^{N} \log \hat{p}_{f,n}^c, \tag{14}$$

where $\hat{p}_{f,n}^c$ represents the probability of ground-truth node category. Moreover, since edges are embedded with the hierarchical relationship between input primitives, such as the connection relationship within the symbol and the spatial relationship between different symbols, we classify each edge according to such relationships:

$$\hat{p}_{b,(i,j)} = \mathrm{softmax}(\mathbf{W}_{b,o}\mathbf{b}^{i,j}), \tag{15}$$

$$\mathcal{L}_{eb} = -\sum_{e_{\mathbf{x}}^{i,j} \in E_{\mathbf{x}}} \log \hat{p}_{b,(i,j)}^c. \tag{16}$$

Decoder Supervision Similarly, the decoder is optimized to predict the target symbol of each node embedding $\mathbf{z}^t$:

$$\hat{p}_{z,t} = \mathrm{softmax}(\mathbf{W}_{z,o}\mathbf{z}^t), \tag{17}$$

$$\mathcal{L}_{nz} = -\sum_{t=1}^{T} \log \hat{p}_{z,t}^c. \tag{18}$$

Furthermore, for the edge in E_{pc}, we embed it as:

$$\mathbf{g}^{pa(t),t} = [\mathbf{z}^{pa(t)}; \mathbf{z}^t]), \tag{19}$$

Then, $\mathbf{g}^{pa(t),t}$ is learned to predict the spatial relationship represented by the corresponding edge in the SLT:

$$\hat{p}_{g,(pa(t),t)} = \mathrm{softmax}(\mathbf{W}_{g,o}\mathbf{g}^{pa(t),t}), \tag{20}$$

$$\mathcal{L}_{eg} = -\sum_{e_{\mathbf{y}}^{pa(t),t} \in E_{pc}} \log \hat{p}_{g,(pa(t),t)}^c. \tag{21}$$

Learning from Sub-graph

While the above technique is able to model the generation of $(v_{\mathbf{y}}^t, E_{\mathbf{y}}^t)$ in Equation 2 with $G_{\mathbf{y}}^{<t}$ and $Attn(G_{\mathbf{x}}, t)$, source node selection via $Attn(\cdot)$ is still uncontrollable, because the vanilla attention is strictly dense and may lack focus on the right symbol, i.e. sub-graph $G_{\mathbf{x}}^{s,t}$, when the input is complex. In this section, we introduce how to learn from $G_{\mathbf{x}}^{s,t}$ to make the recognition more *accurate* and *interpretable*.

Sub-graph Attention The vanilla attention mechanism aligns a distribution α_m^t with softmax to every nodes in $G_{\mathbf{x}}$. However, due to the attribute of softmax function, source nodes that are irrelevant to $v_{\mathbf{y}}^t$ will also be assigned weights greater than zero. Ideally, α_m^t should be a uniform distribution on the related sub-graph $G_{\mathbf{x}}^{s,t}$:

$$p_t(n) = \begin{cases} 1/|V_{\mathbf{x}}^{s,t}|, & v_{\mathbf{x}}^n \in V_{\mathbf{x}}^{s,t} \\ 0, & otherwise \end{cases}. \tag{22}$$

To this end, we propose *Sub-graph Attention* to guide the attention to related sub-graph $G_{\mathbf{x}}^{s,t}$. The generation of sub-graph is elaborated in the next subsection. Specifically, we regularize the attention coefficients output at time step t via:

$$\mathcal{L}_{sa} = -\sum_{n=1}^{N} p_t(n) \log \bar{\alpha}^{t,n}, \tag{23}$$

where $\bar{\alpha}_{t,n}$ indicates the average of the attention coefficients for all the decoder blocks. Furthermore, in order to further enhance the perception of sub-graphs, the fused node embedding of $G_{\mathbf{x}}^{s,t}$ is used to predict the target symbol represented by target node $v_{\mathbf{y}}^t$:

$$\hat{p}_{s,t} = \mathrm{softmax}(\mathbf{W}_{s,o}(\frac{1}{|V_{\mathbf{x}}^{s,t}|}\sum_{v_{\mathbf{x}}^n \in V_{\mathbf{x}}^{s,t}} \mathbf{f}^n)), \tag{24}$$

$$\mathcal{L}_{sg} = -\sum_{t=1}^{T} \log \hat{p}_{s,t}^c. \tag{25}$$

Non-adjacent Sub-graph Masking As can be seen in Fig. 4, for a pair of nodes with a parent-child edge in the target graph, their corresponding symbols are in the adjacent areas of the formula. By constructing $G_{\mathbf{x}}$ with the LOS algorithm, the sub-graphs corresponding to the pair of target nodes are also adjacent, and we can take advantage. Thus, when generating the current node, we *mask* all sub-graphs that are not adjacent to the corresponding sub-graph of its parent node. Intuitively, sub-graphs corresponding to previously parsed nodes are also masked.

Since the label of the correspondence between $G_{\mathbf{x}}^{s,t}$ and $v_{\mathbf{y}}^t$ is not available in the generation phase, we segment $G_{\mathbf{x}}^{s,t}$ for $v_{\mathbf{y}}^t$ via the output attention coefficients of the last decoder block and the edge classification results of the encoder GNN. Specifically, by removing the edges between the source primitives that are classified as connections between two different symbols, we get a set of sub-graphs. Then the sub-graph to which the source node $v_{\mathbf{x}}^n$ aligned with the largest coefficient $\alpha_{\max}^t$ belongs is regarded as the sub-graph $G_{\mathbf{x}}^{s,t}$ for target node $v_{\mathbf{y}}^t$. An example can been in Fig. 5.

Experiment

Implementation Details

The proposed G2G model is trained end-to-end. The final objective function of the model combines the supervision of encoder, decoder, and the sub-graph attention:

$$\mathcal{L} = \lambda_1\mathcal{L}_{nf} + \lambda_2\mathcal{L}_{eb} + \lambda_3\mathcal{L}_{nz} + \lambda_4\mathcal{L}_{eg} + \lambda_5\mathcal{L}_{sa} + \lambda_6\mathcal{L}_{sg}. \tag{26}$$

The coefficients of different supervision losses are set experimentally. Specifically, we set $\lambda_1 = \lambda_2 = \lambda_6 = 0.5$ to impose the same supervision on learning the representations of the nodes, edges and sub-graphs in the input graph. The supervision loss coefficients for the generation of nodes and edges in target graph are set to $\lambda_2 = \lambda_3 = 1$. We set $\lambda_5 = 0.3$ to guide the distribution of attention coefficients on the source sub-graphs.

The proposed model are optimized via the adaptive moment estimation (Adam, Kingma and Ba 2015) with learning rate $5e-4$. Both the decoder and the encoder stack 3 GNN blocks. The network for pre-extracting the input primitive features has 4 blocks. We use 256 for the embedding dimension C' of decoder GNN, and 400 for the dimension C of sub-graph attention. Our models were implemented in PyTorch and optimized on two 12GB Nvidia TITAN X GPUs.

Datasets

We evaluate our model on the large public dataset available from the Competition on Recognition of Online Handwritten Mathematical Expressions (CROHME) (Mouchère et al. 2016). There are 101 math symbol classes and 6 structural relation ("Above", "Below", "Inside", "Superscript", "Subscript", and "Right") in this dataset.

The CROHME training set contains 8,835 formulas with both symbol-level and expression-level annotations, and the test sets for CROHME 2013/2014/2016 contain 671/986/1,147 formulas, respectively. Consistent with participating systems in CROHME, we use the test set of CROHME 2013 as a validation set in training stage, and use the test sets of CROHME 2014 and 2016 to evaluate our proposed model. The performance is measured by ExpRate, defined as the percentage of correctly recognized formulas.

Ablation Study

We conduct benchmarks on CROHME 2014 to analyze the contribution of each component in the proposed G2G model.
Image-to-Markup: We set the image-to-markup system as baseline. The features of handwriting trajectory are fed into a convolutional decoder with pre-aware unit (Wu et al. 2020) to generate the LaTeX format markup. By learning $\mathbf{x} \rightarrow \mathbf{y}$ end-to-end from training data, the "Baseline" achieves expression level accuracy with ExpRate 46.16%. However, relationships between source primitives are still unclear.
Encoder GNN: In the method "+ Encoder GNN", we formulate the input as a graph $G_{\mathbf{X}}$ and further explore the relationship between input primitives with the encoder GNN. We observe that the ExpRate is increased by 1.31%. Moreover, the embedded edges between different primitives can be classified by the encoder GNN. The accuracy of stroke classification ("Node") and determining whether a pair of strokes belong to the same mathematical symbol ("Edge") achieve 91.93% and 99.26%, respectively.
Sub-graph Attention: It is worth noting that annotation tokens like a pair of "$\{\}$" in the LaTeX format markup and the added end child nodes in the modified SLT do not correspond to primitives of the source graph. Thus, "+ Visual Sentinel" (Lu et al. 2017) is used to automatically infer these tokens or nodes. Although the ExpRate is increased by 2.03%, this operation decreases "Node" and "Edge" by 0.09% and 0.15%, respectively.

Method	ExpRate	Node	Edge
Baseline	46.15	-	-
+ Encoder GNN	47.46	91.93	99.26
+ Visual Sentinel	49.49	91.84	99.11
+ Sub-graph Attention	52.13	92.35	99.25
+ Decoder GNN	53.65	92.89	99.33
+ Sub-graph Masking	**54.46**	**92.89**	**99.33**

Table 1: Ablation study of the G2G model on CROHME 2014. "+" denotes to append the current part to the previous system. Accuracies are in $\%$.

Besides, after guiding the decoder to learn the correspondence and semantic connection between the target token and the input sub-graph in the method "+ Sub-graph Attention", the ExpRate, "Node", and "Edge" are increased by 2.64%, 0.51%, and 0.14%, respectively.
Decoder GNN: It explicitly learns the hierarchical structures in the markup with different types of directed edges (i.e. grandparent-to-grandchild, parent-to-child, and brother-to-brother) in the target graph $G_{\mathbf{y}}$. The application of the decoder GNN leads to a further improvement for the recognizer. We observe that after "+ Decoder GNN", the ExpRate, "Node", and "Edge" are increased by 1.52%, 0.54% and 0.08%, respectively.
Non-adjacent Sub-graph Masking: Additionally, during the generation, we exploit the neighbor relationship in $G_{\mathbf{y}}$ and the correspondence between the target nodes in $G_{\mathbf{y}}$ and the sub-graphs of $G_{\mathbf{X}}$ for improvement. In the method "+ Sub-graph Masking", we observe that the ExpRate is further increased by 0.80%. It validates that our proposed G2G model are *accurate* and *interpretable*. Compared with the image-to-markup baseline, our G2G model boosts the performance by 8.31% on CROHME 2014 test set.

Source Graph Representation Learning

Method	Node	Edge	Spatial
GAT (Ye et al. 2019)	88.20	87.72	-
Encoder GNN w/o Spatial Classification	89.91	97.93	-
Encoder GNN	90.80	99.21	96.45
G2G	**92.89**	**99.33**	**97.39**

Table 2: The results of source graph representation learning on CROHME 2014 (in %). "Spatial" indicates to classify edges according to the spatial relationship between the corresponding symbols of two strokes.

We further investigated the performances of different models on source primitive representations and relationships learning. The GAT model proposed in (Ye et al. 2019) can directly classify the contextual primitives in the source graph. We input the same initial features of nodes and edges used by our method into the model and set it as a benchmark. The experimental results are listed in Table 2.

By injecting the embedding of node pairs into the edge between them, the "Encoder GNN" without spatial classification increases the classification accuracies of nodes and edges by 1.71% and 10.21%, respectively. Moreover, we observe that after adding spatial relationship classification task, "Node" and "Edge" are increased by 0.89% and 1.28%, respectively. Interestingly, when end-to-end learning from graph to graph, the ability of representing source graph for the encoder GNN is further enhanced.

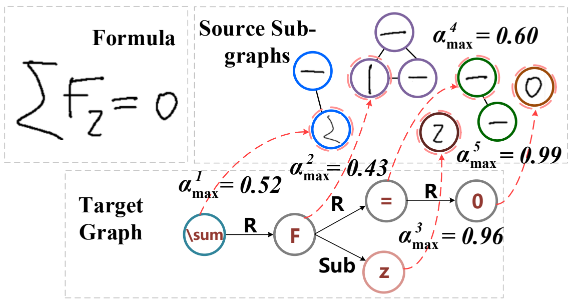

Figure 5: An example of the recognition process. Edges classified as different symbol in the source graph are deleted to obtain the set of sub-graphs. The end child node and edge sets E_{gg} and E_{bb} in the target graph are not displayed.

Symbol Segmentation

One weakness of the image-to-markup method is that symbol corresponding to the target token could not be segmented explicitly, which brings difficulties for post-processing and interpretation. As shown in the Fig. 5, our G2G model resolves this problem by combining the representation learning of source graph and the alignment of sub-graph attention.

System		CROHME 2014		CROHME 2016	
		Segment	Seg+Class	Segment	Seg+class
UPV	G	90.71	84.18	-	-
Tokyo	G	-	-	93.25	87.58
Ours	G2G	**97.34**	**91.79**	**96.96**	**91.05**

Table 3: Symbol level evaluation. The precision metrics for symbol segmentation ("Segment"), and detection of symbols with their correct classification ("Seg+Class") are in %. "G" indicates the grammatical system.

Segmentation results are listed in Table 3, "UPV" and "Tokyo" are the top 1 system using only official training data in the CROHME 2014 and 2016 competition, respectively. The experimental results show the superiority of our proposed G2G model over existing HMER methods in terms of symbol segmentation and explicit structure analysis.

Comparison with the State-of-the-Art

Table 4 shows the results of our proposed G2G model with comparison with participated systems in CROHME and HMER models presented recently. For fairness, we show the results of systems trained on only official data. Besides, systems using additional language corpus, data augmentation, and model ensemble are also removed.

System			CROHME 2014	CROHME 2016
UPV	G	S	37.22	-
Tokyo	G	S	-	43.94
QD-GGA (Mahdavi, Sun, and Zanibbi 2020)	G	S	32.04	32.84
Graphics (Julca-Aguilar et al. 2020)	G	S	-	40.80
WYGIWYS (Deng et al. 2017)	I2M	-	36.41	-
End-to-End (Le and Nakagawa 2017)	I2M	-	35.19	-
WAP (Zhang et al. 2017)	I2M	-	40.40	37.10
PAL (Wu et al. 2018)	I2M	-	39.66	-
DenseMSA (Zhang, Du, and Dai 2018)	I2M	-	43.00	40.00
TAP (Zhang, Du, and Dai 2019)	I2M	S	50.41	-
PAL-v2 (Wu et al. 2020)	I2M	-	48.88	49.61
Tree Decoder (Zhang et al. 2020)	I2M	-	49.10	48.50
Dual Loss Attention (Le 2020)	I2M	-	49.85	47.34
Ours	G2G	S	**54.46**	**52.05**

Table 4: Formula level evaluation of HMER systems on CROHME 2014 and 2016. The metric ExpRate (in %) is the index that ranks the participating systems in the CROHME competitions. "I2M" denotes the image-to-markup system, and "S" indicates that symbol-level annotations were used for training.

We can see from Table 4 that the grammatical method could not automatically leverage the mathematical context and grammatical information contained in the markup to improve its performance, though such methods can explicitly parse the component relationships in the input formula. By learning from graph to graph, our method overcomes such shortcomings of the grammatical method. Besides, compared with the previous state-of-the-art image-to-markup method, our method explicitly learns the hierarchical structures in both input formula and output markup. Our G2G model establishes new SOTAs on both test sets, and also provides explicit symbol-level segmentation. Thus, it is both accurate and interpretable.

Conclusion

In this paper, we propose to formulate HMER as a *graph-to-graph learning* problem to explore the mathematical semantics and hierarchical structures in the formula and its markup. By combining the novel sub-graph attention with graph representation learning, the proposed G2G model could explicitly segment mathematical symbols corresponding to each target node while still being trained end-to-end. Evaluated on both symbol and formula level on the public dataset, our G2G model shows significant improvement compared with previous state-of-the-art image-to-markup and grammatical methods. Experimental results demonstrate that the proposed G2G model enables accurate, interpretable and end-to-end HMER. In the future work, we may extend the proposed G2G to offline HMER by using connected components or detected symbols as the input primitive.

Acknowledgments

This work has been supported by the National Key Research and Development Program Grant 2018YFB1005000, the National Natural Science Foundation of China (NSFC) grants 61773376, 61836014, 61721004, and the Youth Innovation Promotion Association of CAS under Grant 2019141.

References

Alvaro, F.; Sánchez, J.; and Benedí, J. 2014. Recognition of on-line handwritten mathematical expressions using 2D stochastic context-free grammars and hidden Markov models. *Pattern Recognition Letters* 35: 58–67.

Alvaro, F.; Sánchez, J.; and Benedí, J. 2016. An integrated grammar-based approach for mathematical expression recognition. *Pattern Recognition* 51: 135–147.

Anderson, R. H. 1967. 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*, 436–459. ACM.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473.

Chorowski, J.; Bahdanau, D.; Serdyuk, D.; Cho, K.; and Bengio, Y. 2015. Attention-Based Models for Speech Recognition. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, 577–585.

Dauphin, Y. N.; Fan, A.; Auli, M.; and Grangier, D. 2017. Language Modeling with Gated Convolutional Networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 933–941.

Deng, Y.; Kanervisto, A.; Ling, J.; and Rush, A. M. 2017. Image-to-Markup Generation with Coarse-to-Fine Attention. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 980–989.

Gehring, J.; Auli, M.; Grangier, D.; Yarats, D.; and Dauphin, Y. N. 2017. Convolutional Sequence to Sequence Learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 1243–1252.

Hu, L.; and Zanibbi, R. 2016. Line-of-Sight Stroke Graphs and Parzen Shape Context Features for Handwritten Math Formula Representation and Symbol Segmentation. In *15th International Conference on Frontiers in Handwriting Recognition, ICFHR 2016, Shenzhen, China, October 23-26, 2016*, 180–186.

Julca-Aguilar, F. D.; Mouchère, H.; Viard-Gaudin, C.; and Hirata, N. S. T. 2020. A general framework for the recognition of online handwritten graphics. *IJDAR* 23(2): 143–160.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In Bengio, Y.; and LeCun, Y., eds., *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Lavirotte, S. 1997. Optical Formula Recognition. In *4th International Conference Document Analysis and Recognition (ICDAR '97), 2-Volume Set, August 18-20, 1997, Ulm, Germany, Proceedings*, 357–361.

Le, A. D. 2020. Recognizing handwritten mathematical expressions via paired dual loss attention network and printed mathematical expressions. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR Workshops 2020, Seattle, WA, USA, June 14-19, 2020*, 2413–2418.

Le, A. D.; and Nakagawa, M. 2017. Training an End-to-End System for Handwritten Mathematical Expression Recognition by Generated Patterns. In *14th IAPR International Conference on Document Analysis and Recognition, ICDAR 2017, Kyoto, Japan, November 9-15, 2017*, 1056–1061.

Li, L.; Tang, S.; Deng, L.; Zhang, Y.; and Tian, Q. 2017. Image Caption with Global-Local Attention. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, 4133–4139.

Liu, X.; Gao, F.; Zhang, Q.; and Zhao, H. 2019. Graph Convolution for Multimodal Information Extraction from Visually Rich Documents. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 2 (Industry Papers)*, 32–39.

Lu, J.; Xiong, C.; Parikh, D.; and Socher, R. 2017. Knowing When to Look: Adaptive Attention via a Visual Sentinel for Image Captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, 3242–3250.

MacLean, S.; and Labahn, G. 2013. A new approach for recognizing handwritten mathematics using relational grammars and fuzzy sets. *Int. J. Document Anal. Recognit.* 16(2): 139–163.

Mahdavi, M.; Condon, M.; Davila, K.; and Zanibbi, R. 2019. LPGA: Line-of-Sight Parsing with Graph-Based Attention for Math Formula Recognition. In *2019 International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia, September 20-25, 2019*, 647–654.

Mahdavi, M.; Sun, L.; and Zanibbi, R. 2020. Visual Parsing with Query-Driven Global Graph Attention (QD-GGA): Preliminary Results for Handwritten Math Formula Recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR Workshops 2020, Seattle, WA, USA, June 14-19, 2020*, 2429–2438.

Mouchère, H.; Viard-Gaudin, C.; Zanibbi, R.; and Garain, U. 2016. ICFHR2016 CROHME: Competition on Recognition of Online Handwritten Mathematical Expressions. In *15th International Conference on Frontiers in Handwriting*

Recognition, ICFHR 2016, Shenzhen, China, October 23-26, 2016, 607–612.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Wang, J.; Du, J.; Zhang, J.; and Wang, Z. 2019. Multi-modal Attention Network for Handwritten Mathematical Expression Recognition. In *2019 International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia, September 20-25, 2019*, 1181–1186. IEEE.

Wu, J.; Yin, F.; Zhang, Y.; Zhang, X.; and Liu, C. 2018. Image-to-Markup Generation via Paired Adversarial Learning. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2018, Dublin, Ireland, September 10-14, 2018, Proceedings, Part I*, 18–34.

Wu, J.; Yin, F.; Zhang, Y.; Zhang, X.; and Liu, C. 2020. Handwritten Mathematical Expression Recognition via Paired Adversarial Learning. In *Int. J. Comput. Vis.*

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A. C.; Salakhutdinov, R.; Zemel, R. S.; and Bengio, Y. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, 2048–2057.

Yao, T.; Pan, Y.; Li, Y.; and Mei, T. 2018. Exploring Visual Relationship for Image Captioning. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIV*, volume 11218 of *Lecture Notes in Computer Science*, 711–727.

Ye, J.; Zhang, Y.; Yang, Q.; and Liu, C. 2019. Contextual Stroke Classification in Online Handwritten Documents with Graph Attention Networks. In *2019 International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia, September 20-25, 2019*, 993–998.

Zanibbi, R.; and Blostein, D. 2012. Recognition and retrieval of mathematical expressions. *Int. J. Document Anal. Recognit.* 15(4): 331–357.

Zhang, J.; Du, J.; and Dai, L. 2018. Multi-Scale Attention with Dense Encoder for Handwritten Mathematical Expression Recognition. In *24th International Conference on Pattern Recognition, ICPR 2018, Beijing, China, August 20-24, 2018*, 2245–2250.

Zhang, J.; Du, J.; and Dai, L. 2019. Track, Attend, and Parse (TAP): An End-to-End Framework for Online Handwritten Mathematical Expression Recognition. *IEEE Trans. Multimedia* 21(1): 221–233.

Zhang, J.; Du, J.; Yang, Y.; Song, Y.-Z.; Wei, S.; and Dai, L. 2020. A Tree-Structured Decoder for Image-to-Markup Generation. In *Proceedings of the 37nd International Conference on Machine Learning, ICML 2020*, volume 95, 93.

Zhang, J.; Du, J.; Zhang, S.; Liu, D.; Hu, Y.; Hu, J.; Wei, S.; and Dai, L. 2017. Watch, attend and parse: An end-to-end neural network based approach to handwritten mathematical expression recognition. *Pattern Recognition* 71: 196–206.