Stroke-Level Graph Labeling with Edge-weighted Graph Attention Network for Handwritten Mathematical Expression Recognition

Yejing Xie, Harold Mouchère

Stroke-Level Graph Labeling with Edge-weighted Graph Attention Network for Handwritten Mathematical Expression Recognition

Yejing Xie[0009−0002−8360−4696] and Harold Mouchère[0000−0001−6220−7216]

Laboratoire des Sciences du Numérique de Nantes (LS2N), Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, F-44000 Nantes, France
{yejing.xie,harold.mouchere}@univ-nantes.fr

Abstract. Handwritten Mathematic Expression Recognition (HMER) algorithms with deep learning aproaches have developed rapidly in recent years, most algorithms are dependent on heavy pre-training and also complex network structures. Existing architectures are build on encoder-decoder from on-line or off-line inputs to produce LaTeX markup strings, or stroke-level graphs to generate symbol-level graphs. They all remain on a latent space, which is not directly related to the input data: the strokes. Using the Stroke Label Graph modelisation allows a direct connection between the input data and the output labels. In this research, we proposed a novel stroke-level graph labeling method with edge-weighted graph attention network (EGAT). This lightweight model doesn't rely on any pre-training, abandons the laborious process of encoder-decoder, is totaly end-to-end, directly accomplishes stroke-to-stroke feature extraction, and produces strokes and relations classification. Experiments show that our proposed EGAT algorithm can effectively fuse the node features as well as the weighted edge features, and predict the node and edge attributes simultaneously.

Keywords: Handwritten Mathematical Expression Recognition · Graph-based approaches · Graph Attention Network · Stroke Level Labeling

1 Introduction

Mathematical Expression (ME) is an essential part of scientific researching, engineering development, basic education and so on. Compared to the more regular and complicated click-based ME editing tools or markup language (such as LaTeX) for inputting an ME, the handwritten mathematical expression (HME) is more convenient for human editing, but more difficult for machines to recognize due to different writing styles and habits.

Handwritten Mathematical Recognition (HMER), which means transforming handwriting into mark-up information for convenient computer calculation or rendering, is a challenging and promising subject with a variety of possible applications.

Compared to the Optical Character Recognition (OCR) problem, handwritten manuscript recognition is more difficult due to the diversity of manuscript styles. HMER has to confront the general challenges of handwriting recognition, and also the specific dificulty of 2D mathematical expression structure.

According to the different processing objective, general HMER can be divided into online HMER and offline HMER. Online means several strokes with temporal trajectory forming a HME while offline means a static image of HME.

Existing Deep Learning architectures for HMER are usually built on encoder-decoder from on-line or off-line inputs to produce LaTeX markup strings. These sequence-to-sequence models do not take advantage of the graph structure of mathematical layout, the hidden information of the relations between symbols is difficult to be captured and utilized. There are some other algorithms that utilize graph structures or tree structures to recognize as assistive information. However, all of them remain on a latent space, which is not directly related to the input data: the seperate strokes.

In that way, we try to explore the graph based modelisation of HME, and use the modelised data for end-to-end stroke-level recognition.

The contributions of this paper are as follows:

- We proposed a general stroke-level graph labeling architecture for HMER, which can directly accomplish stroke-to-stroke feature extraction, and produce strokes and relations classification. The overview structure is shown in Fig. 1.
- We proposed a stroke level graph modelisation with Line-of-Sight graph structure, and also the node and edge features extraction with Fuzzy Relative Positioning Template (FRPT).
- An edge-weighted graph attention network (EGAT) is proved to effectively fuse the node features as well as the weighted edge features, and predict the node and edge attributes simultaneously.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 introduces the proposed stroke-level graph modelisation method. Section 4 presents the proposed end-to-end for stroke-level graph labeling. Section 5 presents the experimental results.

The open-source code and trained model will be available.[1]

2 Related Work

With the development of HMER algorithms recently, the sequential methods, grammer integrated methods and other tranditional recognition methods are gradually replaced by the more popular end-to-end approches. In recent researches, with the help of high-performance backbone deep networks, the deep information of HME is learned and extracted by different forms of modeling and relabeling method.

[1] https://gitlab.univ-nantes.fr/E19B516G/egat

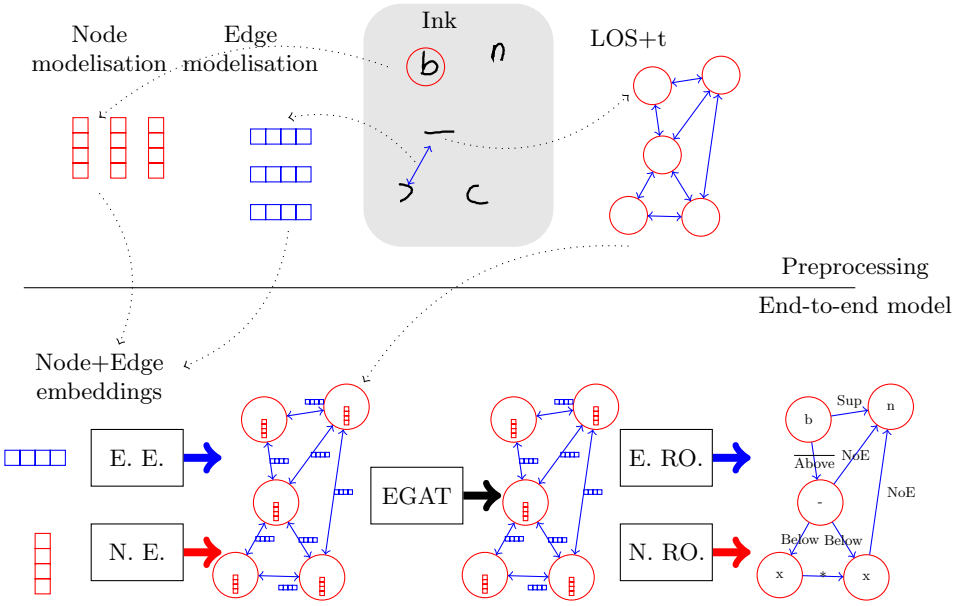


Fig. 1: Overview of the architecture. The preprocessing extracts the node and edge features from the raw stroke coordinates, and the LOS graph augmented by time connections. The Node and Edge Embedding models embed the node and edge features into a high-dimensional vector. The Edge-weighted Graph Attention Network (EGAT) is used to update the node and edge features. The two Readout networks are used to make the final decisions on edges and nodes.

Except the graph-based methods, some researches applied encoder-decoder structure to encode the HME images or sequential data into high-dimensional vectors, and then decode the vectors into LaTeX strings as the final recognition results. Among these methods, bidirectional decoder [38], constrained attention decoder [30], symbol counting vector [10] and other methods are proposed to improve the performance of typical encoder-decoder.

Graph structure is powerful for Mathematical Expression (ME) modelisation, which can better represent the deep information between symbols. Thus, more and more researches have been proposed to use graph structure for HMER, no matter tranditional methods or deep learning methods. Since our research focuses on graph, graph modelisation and applications of graph modelisation in HMER, we will pay more attention to the state of the art of "graph-based" HMER methods.

Maximum Spanning Trees (MSTs). Some early tranditional methods [3, 6, 14] have taken advantage of graph structure by using Maximum Spanning Tree to generate graph structure of ME. These traditional MST-based approaches are no longer dominant after the popularity of machine learning in recent years. How-

ever, this most elementary approach may also provide guidance and inspiration for sophisticated deep learning networks, which is the most in trend nowadays.

Graph with Contextual Information. A number of instances [1, 4, 24] have shown that graphical structure information can be specified by pre-designed graph grammar, which can be employed for HMER. F. Julca-Aguilar et al. [7–9] proposed the graph-driven general frameworks for online HMER, with data-driven hypotheses prediction and graph grammar parser for parse tree generation. Although the graph based contextual information algorithms can parse equations efficiently, the complex manual grammar designing work is prohibitively expensive.

Tree Based Decoder. A tree based decoder is a more flexible and efficient way to decode the structural information and represent the deep information of ME. The first trial [37] extended chain-structured BLSTM as tree structure decoder for online HMER. While [28] proposed an online HMER algorithm by BLSTM-based symbol-relation temporal classifier and handmade a series of path extraction rules for tree reconstruction. A sequential relation decoder [36] was proposed to decode the encoded online trajectory into tree structure. And the same team proposed a tree structure decoder with similar principle for offline HMER [22].

However, these tree based decoders always convert tree structure by a fixed order, thus they fail to take full advantage of the diverse expressions of tree. To improve on this point, C. Wu et al. [31] proposed a novel tree decoder (TDv2) to make maximum advantage of tree structure labels. These algorithms merely consider structural information and lack the overall representation of grammatical information in deep learning features. Y. Yuan et al. proposed a Syntax-Aware Network (SAN) [35], first applied syntax information into the encoder-decoder structure with help of syntax tree. Influenced by visual parsing, M. Mahdavi et al. [12] adopted visual parsing, especially graph parsing for HMER and proposed Query-Driven Global Graph Attention (QD-GGA) algorithm.

Graph Neural Network: In recent years, there has been rising enthusiasm for the extension of deep learning methods to graphs, which is Graph Neural Network (GNN) [25]. GNN is used to handle the HMER problem in recent some research, which can effectively perform message passing by the graph representation of equation.

J. Wu et. all [32] proposed a Graph-to-Graph model for online HMER with GAT based encoder-decoder structure, which is a symbol level labeling approche. J. Tang et. all [27] proposed an offline HMER algorithm with Graph Reasoning Network (GRN). On the basis of all this work, J. Tang et. all [26] has presented a new offline HMER algorithm with Graph Neural Network encoder and Transformer decoder.

So far, there isn't any research that directly uses the stroke-level graph modelisation and labeling for HMER, cannot fully utilize the features of each stroke separately. In this paper, we provide a new orientation for solving the HMER problem.

3 Stroke-level Modelisation

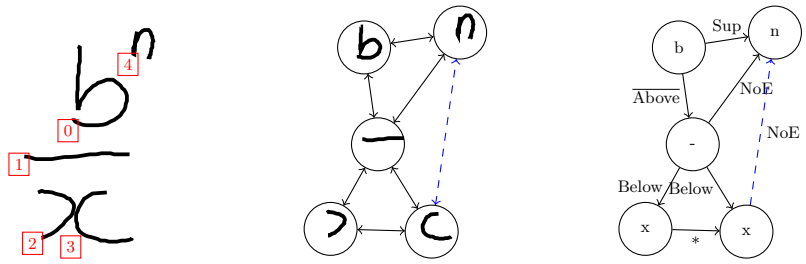
Graph construction serves as the foundational step in the development of a Graph based Handwritten Mathematical Expression Recognition (HMER) system. In this context, graph construction requires the detailed modeling of the node features, edge features, and the connectivity relationships between them.

In this study, Nodes represent the individual strokes of the Online Handwritten Mathematical Expression(OHME), while Edges represent the relations between 2 strokes. The aim of the Nodes Modelisation is that each node can effectively represent the shape features of the corresponding stroke. And the Edge Modelisation can well represent the temporal, spatial, and other high dimension relations between 2 strokes.

Such "Stroke Level" graph construction can directly represent and preserves the Online stroke information, which is better adapted than the "Symbol Level" graph construction for end-to-end networks.

3.1 Graph Construction with Line-of-Sight

We applied the Line-of-Sight (LOS)[5] method to construct the connected relations between strokes. The main idea is whether the strokes will be still visible due to the occlusion of other strokes. Based on LOS, we also add the temporal order connections between strokes to increase the connectivity of the graph, which noted as LOS+t graph. From here, we only consider the visible relations between strokes, without the consideration of attributes on the edges. As an instance, OHME "$\frac{b^n}{x}$" contains 4 strokes as shown in Fig 2a, while the undirected graph constructed by LOS+t is shown in Fig 2b.



(a) Ink (b) LOS+t (c) SLG

Fig. 2: From OHME to Stroke Label Graph(SLG). a) The OHME "$\frac{b^n}{x}$" with 5 strokes, numbers show the time order. b) The LOS+t graph, visibility between strokes correspond to plain edges and time connections are in dashed. c) The ground-truth Stroke Label Graph, NoE states for No-Edge, and "*" label means segmentation edge.

3.2 Node and Edge Modelisation

After the construction of graph structure with LOS, Node and Edge Modelisation are required to represent the shape features of strokes and the relations between strokes according to online stroke coordinate information. These steps can be seemed as pre-feature extraction before input to an end-to-end deep neural network, which reduces the hardness of the following model training.

Node Modelisation Only the x and y coordinates of each stroke is utilised to be directly modelised as the node features. However, due to the diversity of data collection devices and the different writing habits of users, there are large differences in the stroke coordinates of the same symbol class. Therefore, the standardization is necessary for reducing these negative impacts. We applied a strategy of removing writing speed [21] for the raw stroke coordinates, and also the gaussian normalization in stroke level. Both these approches are used for containing the stroke shape information and also reducing the influence of writing speed and writing habits.

Edge Modelisation The edge modelisation is more complicated than the node modelisation, which requires a strategy for modeling or extracting the relations features between strokes. The previous online documents analysis works extracted the multi-dimensional geoetric features of bounding boxes of strokes, such as [34].

Rather than simply considering the geometric relationship between the strokes bounding box, in order to better capture the shape and bi-directional position information. In this study, we applied a Fuzzy Relative Positioning Template (FRPT) [2] for relations extraction.

The main idea is to calculate the radian degree between standard vectors $\overrightarrow{e}$ in 4 directions and the vector $\overrightarrow{OP_i}$ from the center of initial stroke O to the sampling points of target stroke P_i, as shown in Eq. 1. Only the features of vectors that make an acute angle with the standard vector are preserved. While the directed edge of the initial node pointing to the target node retains the shape information of the target node.

$$\theta_i = max\left(0, 1 - \frac{2}{\pi}\arccos\left(\frac{\overrightarrow{OP_i}\cdot\overrightarrow{e}}{\sqrt{|\overrightarrow{OP_i}|^2 + |\overrightarrow{e}|^2}}\right)\right) \tag{1}$$

Therefore, the exchange of initial and target stroke will lead to different results, with a visualized example in Fig 3. In the first row, the initial stroke is "*b*" and the target stroke is "*n*", while the second row is the opposite. The lighter pixel means larger embedding value, and the darker means closer to 0. The connections between the nodes always have 2 directions with different embeddings.

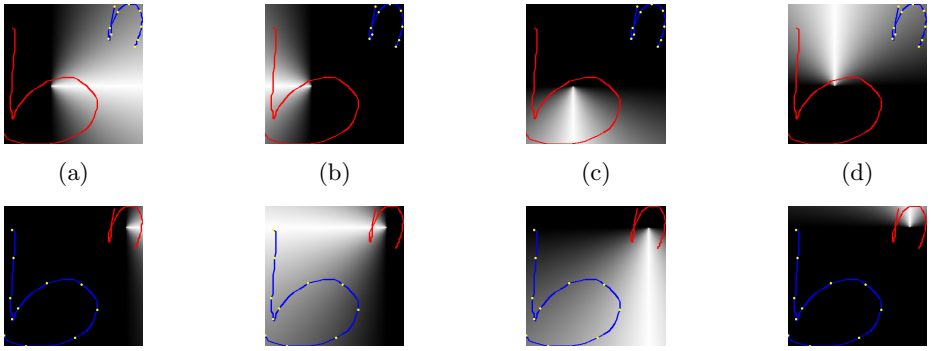


Fig. 3: Fuzzy Relative Positioning Template for b to n and n to b in 4 directions.

3.3 Ground Truth Relabeling with Graph Structure

The ground truth of OHME can be represented as different formats, such as MathML tree, LaTeX string, and also the stroke-level graph called Stroke Label Graph (SLG) proposed by [19, 16, 18, 17, 15, 33, 13]. Take the advantage of stroke-level graph based end-to-end network and also the previous OHME graph modelisation, we can apply the SLG as ground truth in following model design and training process with only several simple edge modifications.

There are 101 classes of symbols and 6 relations in original SLG of CROHME datasets. Except the "Right", "Sup", "Sub", "Above", "Below" and "Inside" relations between strokes. These 6 relations are all directional according to the relative position of the strokes. In order to avoid the ambiguity of directions, we add an opposite directed edge for every edge in SLG, these opposite relations are noted as "$\overline{\text{Right}}$", "$\overline{\text{Sup}}$", "$\overline{\text{Sub}}$", "$\overline{\text{Above}}$", "$\overline{\text{Below}}$" and "$\overline{\text{Inside}}$". After that, we keep only one directed edge for each pair of connected strokes according to the writing order, which means every edge in the graph are directed from the early written down stroke to the later stroke. This strategy preserves the graph structure of SLG and ignores syntactic rules, but avoids confusion of directions through temporal ordering.

In addition, we add an "*" relation that signifies the 2 strokes belong to one symbol, and a "NoE" means there is no relation between 2 strokes. The edited SLG example is shown as Fig 2c.

In conclusion, the ground truth of OHME can be represented as a stroke-level graph with 101 classes of symbols and 14 relations. So we convert the complex OHMER problem into a graph labeling problem with $C_{node} = 101$ classes of node classification and $C_{edge} = 14$ classes of edge classification.

4 End-to-end Model

In this section, we will introduce the details of proposed deep learning model with edge-weighted graph neural network.

4.1 General Structure

The general structure of end-to-end model is shown in Fig 1, which could be divided into 3 parts: Graph embedding, Edge-weighted Graph Attention Network and Readout. The entire model is an end-to-end trainable network, which can be trained with the help of relabeled ground truth graph structure and stroke-level OHME Graph Modelisation as input structure.

4.2 Graph Embedding Network

The graph embedding relies on the result of OHME modelisation, which extracts the deep features of node and edge separately. Since the node represents each stroke, intuitively we would like to preserve more of the shape features of the strokes. While the edge represents the relations between 2 strokes, we would like to preserve more features like position and directions. The efficient graph embedding will give a positive influence on the following features fusion message passing and classification.

Node Embedding Network The node information which represents a stroke contains a series of coordinates by time order and already pre-processed, as shown in Eq. 2, the input $i-th$ node features h_i^{input} is a $N \times 2$ matrix, where N is the number of sampling points.

$$h_i^{input} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\} \in \mathbb{R}^{N \times 2} \tag{2}$$

This temporal sequence can be embedded by Recurrent Neural Network (RNN), such as LSTM or GRU, as well as Transformer Encoder. But the shape information of the strokes is more important, so we tried a 1-D convolutional deep network XceptionTime [23] that was more capable of capturing the shape information. As shown in Eq. 3, the i^{th} embedded node features h_i^{emb} is calculated by node embedding network $F(x)$, where we employed XceptionTime in this study. The node embedding network embedded the node features into a H_{node}^{emb} dimension vector.

$$h_i^{emb} = F(h_i^{input}) \in \mathbb{R}^{H_{node}^{emb}} \tag{3}$$

XceptionTime is known for its depth-wise separable convolution, which can reduce computation complexity and also keep the sequence information. Subsequent experiments will verify that XceptionTime is the most accurate node embedding network.

Edge Embedding Network With the Edge Modelisation strategy FRPT in Eq. 1, the edge input features $b_{ij}^{input} \in \mathbb{R}^{4N'}$ between node i and j is a $4N'$ vector, where N' is the number of sampling points and is calculated in 4 directions.

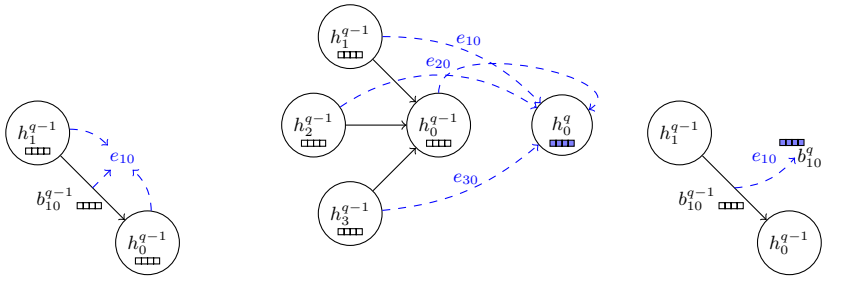
$$b_{ij}^{emb} = G(b_{ij}^{input}) \in \mathbb{R}^{H_{edge}^{emb}} \tag{4}$$

Although the edge modelisation is more complicated than the node modelisation, the edge embedding needs less precise information than the node embedding. Consequently, we employed a simple Multi layer Perception (MLP) $G(x)$ to embed the edge features into a H_{edge}^{emb} dimension vector.

4.3 Edge-weighted Graph Attention Network

With the excellent graph structure modeling capability and also the efficient relations and dependencies caption ability, Graph Neural Network can be employed successfully in well-modeled OHME. In addition to the stroke information embedded in nodes, the relations between nodes embedded in edges are also important for OHME labeling. However, edge information can not be represented as reachability or a simple weight, and is supposed to be dynamically involved in message passing and features update.

Thus, we propose an edge-weighted graph attention network (EGAT) with the help of graph attention mechanisms [29], which can simultaneously update the node features and also the edge features.



(a) Attention weight computation.

(b) Node embedding h update.

(c) Edge embedding b update.

Fig. 4: Illustration of the Edge-weighted Graph Attention Network (EGAT).

For every layer, when we calculate the attention weights, except the concatenation of corresponding neighbor node features, we also concatenate the edge features together by Eq. 5. $W_h \in \mathbb{R}^{H_{node}^{q-1} \times H_{node}^{q}}$ and $W_b \in \mathbb{R}^{H_{edge}^{q-1} \times H_{edge}^{q}}$ are learnable weights for node features and edge features, while $a \in \mathbb{R}^{2H_{node}^{q}+H_{edge}^{q}}$ is a learnable attention coefficient, where H_{node}^{q-1} and H_{edge}^{q-1} are the dimensions of node features and edge features in the $(q-1)^{th}$ layer, and H_{node}^{q} and H_{edge}^{q}

are the dimensions of node features and edge features in the q^{th} layer.

$$e_{ij} = a\left[W_h \cdot h_i \parallel W_h \cdot h_j \parallel W_b \cdot b_{ij}\right] \tag{5}$$

Then, with the help of softmax function, we can get the attention weights $\alpha_{ij} \in \mathbb{R}^{H^{q-1}_{node} \times H^{q}_{node}}$ by Eq. 6.

$$\alpha_{ij} = softmax(e_{ij}) = \frac{exp(e_{ij})}{\sum_{k \in \mathcal{N}_i} exp(e_{ik})} \tag{6}$$

In the q-th layer, the node i features $h^q_i \in \mathbb{R}^{H^q_{node}}$ are updated by all the neighbor node features in last layer $h^{q-1}_i \in \mathbb{R}^{H^{q-1}_{node}}$ while $j \in \mathcal{N}$ and also the corresponding attention weights α_{ij}. The message passing will be achieved through the weighted summation of neighbor information, by Eq. 7.

$$h^q_i = \sum_{j \in \mathcal{N}_i} \left(\alpha_{ij} \cdot W_h \cdot h^{q-1}_j\right) \tag{7}$$

Likewise, the edge features $b^q_{ij} \in \mathbb{R}^{H^q_{edge}}$ will also be updated by the corresponding attention weights α_{ij} and the edge features in last layer $b^{q-1}_{ij} \in \mathbb{R}^{H^{q-1}_{edge}}$ by Eq. 8.

$$b^q_{ij} = \alpha_{ij} \cdot W_b \cdot b^{q-1}_{ij} \tag{8}$$

To summarize the EGAT, as described in Eq. 9. In case we use multi-batch training with batch size B, the input of q^{th} layer will be: $h^{q-1} \in \mathbb{R}^{B \times H^{q-1}_{node}}$, $b^{q-1} \in \mathbb{R}^{B \times B \times H^{q-1}_{edge}}$ and adjacency matrix $\mathcal{A} \in \mathbb{R}^{B \times B}$.

$$h^q, b^q = \mathcal{G}(h^{q-1}, b^{q-1}, \mathcal{A}) \tag{9}$$

After EGAT features fusion and message passing network $\mathcal{G}(x)$, the output of q^{th} layer node features $h^q \in \mathbb{R}^{B \times H^q_{node}}$ and edge features $b^q \in \mathbb{R}^{B \times B \times H^q_{edge}}$ will be updated. In addition, the LOS based adjacency matric $\mathcal{A}$ isn't altered as a result.

4.4 Readout

After multi-layer EGAT, the fused node and edge features will be entered into their individual Readout networks for the final classification. Both Readout networks for node features and edge features are MLPs with the similar structure.

Due to the ground truth is edited SLG with directed edges according to the writing order, we concatenate the bi-directional edges b_{ij} and b_{ji}, and then input the concatenated edge features into the Edge Readout network.

$$h^{out} = \mathcal{R}(h^q) \in \mathbb{R}^{B \times C_{node}} \tag{10}$$

$$b^{out} = \mathcal{R}(b^q) \in \mathbb{R}^{B \times B \times C_{edge}} \tag{11}$$

4.5 Node and Edge Loss Function

To achieve the node and edge classification, we employed the cross entropy loss function for node classification, and focal loss [11] for edge classification. The no relation edges"NoE" take up a large proportion of the total edges, which will lead to the problem of unbalanced data distribution. Thus, the focal loss can be more effective than the cross entropy loss function. We combined the node loss and edge loss together as the final loss function for the bi-objective optimization.

$$\mathcal{L} = \lambda_1 \mathcal{L}_{node} + \lambda_2 \mathcal{L}_{edge} \tag{12}$$

5 Experiment

5.1 Implementation Details

Our proposed model is implemented by PyTorch-Lightning framework, and trained on a single NVIDIA GeForce RTX 2080 Ti GPU with 12GB memory. All the hyperparameters are selected by finetuning with optuna framework: input node dimension $N = 150$, sampling edge dimension $N' = 10$, node embedding dimension $H^{emb}_{node} = 384$, edge embedding is a 3-layer MLP $G(x) = [40, 256, 384]$, 4-layer EGAT with $H_{node} = [384, 512, 384, 512]$ and $H_{edge} = [384, 256, 512, 256]$, node Readout network is a 3-layer MLP $\mathcal{R}_{node}(x) = [384, 512, 101]$, edge Readout network $\mathcal{R}_{edge}(x) = [512, 384, 14]$, node loss weight $\lambda_1 = 0.4$, edge loss weight $\lambda_2 = 0.6$. The Adam optimizer is used with a learning rate of 0.0004 and a batch size of 256.

For the proposal of multi batch training, all the expressions are unified to sub-expressions with 8 strokes, the sub-expressions are padded with 0 if the number of strokes is less than 8. Thus, each batch contains 32 sub-expressions and 256 strokes in total, which is considered as a graph with 256 nodes. Only inner sub-expression has edge connection, there is no connection between different sub-expression. This strategy is similar to the multi-patch training in computer vision, which can significantly improve ability of generalization in practice and reduce GPU memory usage.

In the training and validation stages, we masked the incomplete symbols from cutting, as well as the padding symbols, to reduce the negative impact of abnormal data on the model. While in the test stage, we used the entire expression as the input for the complete recognition.

5.2 Dataset

The proposed model is trained and evaluated on the newest Competition on Recognition of Online Handwritten Mathematical Expression dataset (CROHME 2023) [33]. The CROHME 2023 provides large-scale on-line OHME data with expressions-level, symbol-level, and stroke-level annotations. There are 101 math symbol classes and 6 structural relation classes ("Right", "Sup", "Sub", "Above", "Below", "Inside").

We did the ablation study on the CROHME 2023 and 2019 test set in stroke-level, but restricted by the fact that stroke-level recent studies are rare, and the training and validation data we used has contained the test 2012 test 2013 test 2014 and test 2016.

The composition of the utilised data is shown in Table 1.

Table 1: CROHME 2023 Dataset.

Dataset	Composition	Number of HME
Train Set	Train2014 Test2013 Test2012 Test2014 Train2023	12024
Validation Set	Test2016 Val2023	1072
Test Set	Test2019 Test2023	3499

5.3 Results

Node Embedding Network This sub-experiment is to verify the effectiveness of the node embedding network, and the results will guide the following entire model training.

We applied different time series Neural Network with framework Tsai[20] for stroke-level classification, including XceptionTime, Bi-LSTM, InceptionTime, Xresnet1D18, ResNet (1D), Transformer Model and RNN-FCN. As shown in Table 2, XceptionTime is the most accurate node embedding network, which is a CNN-based convolutional network for time series data. Unexpectedly, the Transformer Model is not suitable in this case, which is a self-attention based network for sequence data.

Table 2: Node Embedding Network.

Method	Testset 2019		Testset 2023	
	acc. %	prec.%	acc.%	prec.%
XceptionTime	**60.48**	**59.87**	**60.23**	**59.66**
Bi-LSTM	57.94	57.83	58.09	57.99
InceptionTime	56.10	54.32	56.24	54.12
Xresnet1D18	55.88	54.66	55.34	54.29
ResNet (1D)	54.25	54.11	54.36	54.43
TransformerModel	40.70	40.53	40.56	40.45
RNN-FCN	40.19	39.98	40.43	40.12

Edge Modelisation Method We employed Fuzzy Relative Positioning Template (FRPT) for edge modelisation, and compared with tranditional edge modelisation method with 20 dimension geometric features (Geo.) which was utilised in [32]. The used edge embedded network is a simple MLP with 3 layers, same hyperparameters as the end-to-end model. The classification results are 14 classes of edge relations, accurancy and precision shown in Table 3.

Experiments show that FRPT algorithm is more accurate than normal geometric features, which is a more suitable edge modelisation method for OHME.

Table 3: Edge Modelisation Method, with same embedded network and classes as the end-to-end model.

Method	Testset 2019		Testset 2023	
	acc.%	prec.%	acc.%	prec.%
FRPT	**70.55**	**66.92**	**70.69**	**67.01**
Geo.	60.57	51.67	61.21	53.20

Ablation Study A series of ablation studies are conducted to verify the effectiveness of the proposed model in Table 4. We utilised accurancy (acc.) and precision (prec.) to evaluate the segmentation (Seg.), segmentation with classification (Seg. + Class.) and relation classification (Relation) performance in HMER.

Node and edge modelisation and embedding network were trained separately without message passing and features fusion with GNN, which is utilised as baseline the proposed stroke-level modelisation model to compare with following ablation studies.

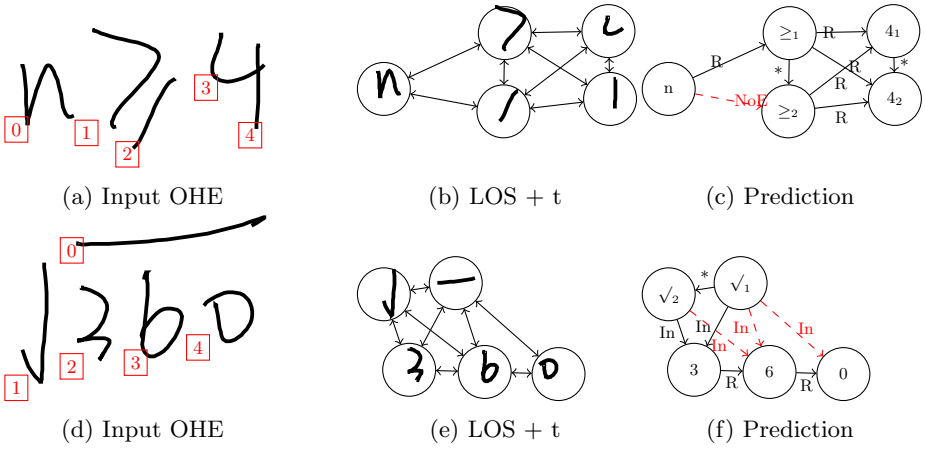
We applied the normal Graph Attention Network (GAT) for message passing only between strokes, without any features fusion of edge information. There is no significant improvement over baseline, due to the lack of relation information.

The incorporation of the EGAT makes a significant improvement in baseline. No matter which node embedding network or edge modelisation method is applied, the segmentation, stroke classification and relation classification accurancy are all improved over 20%. This is a strong indication that the node and edge features can guide and improve each other take the advantage of EGAT.

By comparing different node embedding and edge modelisation methods, the most effective end-to-end model use the XceptionTime for node embedding with FRPT edge features. While Bi-LSTM is also effective especially in testset 2019, but with larger parameter volume than XceptionTime.

Table 4: Ablation Study.

Method			Testset 2019					
E. Mod.	N. Emb.	GNN	Seg.		Seg. + Class.		Relation	
			acc.	prec.	acc.	prec.	acc.	prec.
FRPT	Xcept.	×	71.99	65.23	60.48	59.87	70.55	66.92
Geo.	Xcept.	×	65.23	60.76	60.48	59.87	60.57	61.21
×	Xcept.	GAT	-	-	61.69	61.21	-	-
Geo.	Xcept.	EGAT	93.17	92.01	82.95	85.97	87.99	87.24
FRPT	Trans.	EGAT	93.16	90.21	73.41	75.33	85.77	87.15
FRPT	BiLSTM	EGAT	96.06	**96.84**	**85.41**	87.32	**91.10**	**92.47**
FRPT	**Xcept.**	**EGAT**	**96.18**	95.88	85.07	**88.97**	90.87	90.93
Method			Testset 2023					
E. Mod.	N. Emb.	GNN	Seg.		Seg. + Class.		Relation	
			acc.	prec.	acc.	prec.	acc.	prec.
FRPT	Xcept.	×	72.11	66.38	60.23	59.66	70.69	67.01
Geo.	Xcept.	×	66.09	60.88	60.23	59.66	61.21	53.20
×	Xcept.	GAT	-	-	62.06	61.67	-	-
Geo.	Xcept.	EGAT	93.77	92.10	84.35	84.77	88.88	87.51
FRPT	Trans.	EGAT	93.72	89.24	75.89	77.62	85.31	86.60
FRPT	BiLSTM	EGAT	96.33	96.28	87.10	88.48	90.88	92.03
FRPT	**Xcept.**	**EGAT**	**96.70**	**95.85**	**87.39**	**88.39**	**90.93**	**92.28**



(a) Input OHE (b) LOS + t (c) Prediction

(d) Input OHE (e) LOS + t (f) Prediction

5.4 Cases Analysis

According to the results of experiments, we can see that the proposed model is effective in stroke-level recognition both in stroke and relation classification.

However, there are some misclassification errors that can be fixed by post-processing approches, such as the example in Fig 5d. The node classification are all correct, only one edge classification between the first stroke and the third

stroke is wrong, which is a "Right" relation, but the model predicted it as a "NoE". The second stroke and third compose the "≥" symbol, and the symbol segmentation as well as relation classification between the first stroke and the second stroke are all correct, so we can add the "Right" relation between the first stroke and the second stroke to fix the error.

In several situations, the model will give results that match the spatial location but not the grammar of expression. Such as the expression "$\sqrt{360}$", only the first symbol "3" below the "$\backslash sqrt$" has the "Inside" relation, "6" and "0" have both the "Right" relation with their previous symbol. But the model predicted the extra "Inside" relation between "6" and "$\backslash sqrt$", "0" and "$\backslash sqrt$", which is raisonable in spatial location but not correct in stroke-level prediction.

The interpretability of the model makes a number of reasonable improvements as possible.

6 Conclusion

In this study, we proposed a novel end-to-end stroke-level labeling model for Online Handwritten Mathematical Expression (OHME) based on Edge-weighted Graph Attention Network (EGAT), which is a radically new approach to solving the HMER problem in stroke-level. We have proved the proposed stroke-level modelisation model is effective for stroke and relation classification, and also the positive influence of node and edge features fusion with EGAT. However, in expression level, the accurancy is not quite at SOTA level, and the model is not interpretable enough for the grammar of expression.

In the future, we will focus on the following aspects to improve the baseline model:

- **More essential features for node and edge modelisation.** The preprocessing methods for node modelisation and FRPT for edge modelisation remain potential for improvement. Such as the pre-fusion features of strokes and the distance information for FRPT.
- **Post-processing with simple grammar rules.** After the end-to-end model, a post-processing with simple interpretable grammar rules can be employed to fix the misclassified errors. Such as unify the attributes of the strokes in the same symbol, and the relations between the strokes in the same symbol.
- **Fine-grained edge classification.** Similar positional relations are classified completely different due to the rules of the syntax tree. Fine-grained classification of edges, with the help of syntactic tree rules, will be helpful to improve the accuracy of edge classification

References

1. Álvaro, F., Sánchez, J.A., Benedí, J.M.: An integrated grammar-based approach for mathematical expression recognition. Pattern Recognition **51**, 135–147 (2016)

2. Delaye, A., Anquetil, E.: Fuzzy relative positioning templates for symbol recognition. In: 2011 International Conference on Document Analysis and Recognition. pp. 1220–1224. IEEE (2011)
3. Eto, Y., Suzuki, M.: Mathematical formula recognition using virtual link network. In: Proceedings of sixth international conference on document analysis and recognition. pp. 762–767. IEEE (2001)
4. Han, F., Zhu, S.C.: Bottom-up/top-down image parsing by attribute graph grammar. In: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1. vol. 2, pp. 1778–1785. IEEE (2005)
5. Hu, L., Zanibbi, R.: Line-of-sight stroke graphs and parzen shape context features for handwritten math formula representation and symbol segmentation. In: 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR). pp. 180–186. IEEE (2016)
6. Hu, L., Zanibbi, R.: Mst-based visual parsing of online handwritten mathematical expressions. In: 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR). pp. 337–342. IEEE (2016)
7. Julca-Aguilar, F.: Recognition of online handwritten mathematical expressions using contextual information. Ph.D. thesis, Université de Nantes; Université Bretagne Loire; Universidade de São Paulo (2016)
8. Julca-Aguilar, F., Mouchère, H., Viard-Gaudin, C., Hirata, N.S.: Top-down online handwritten mathematical expression parsing with graph grammar. In: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 20th Iberoamerican Congress, CIARP 2015, Montevideo, Uruguay, November 9-12, 2015, Proceedings 20. pp. 444–451. Springer (2015)
9. Julca-Aguilar, F., Mouchère, H., Viard-Gaudin, C., Hirata, N.S.: A general framework for the recognition of online handwritten graphics. International Journal on Document Analysis and Recognition (IJDAR) **23**, 143–160 (2020)
10. Li, B., Yuan, Y., Liang, D., Liu, X., Ji, Z., Bai, J., Liu, W., Bai, X.: When counting meets hmer: counting-aware network for handwritten mathematical expression recognition. In: European Conference on Computer Vision. pp. 197–214. Springer (2022)
11. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)
12. Mahdavi, M., Zanibbi, R.: Visual parsing with query-driven global graph attention (qd-gga): preliminary results for handwritten math formula recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 570–571 (2020)
13. Mahdavi, M., Zanibbi, R., Mouchère, H., Viard-Gaudin, C., Garain, U.: Icdar 2019 crohme+ tfd: Competition on recognition of handwritten mathematical expressions and typeset formula detection. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 1533–1538. IEEE (2019)
14. Matsakis, N.E.: Recognition of handwritten mathematical expressions. Ph.D. thesis, Massachusetts Institute of Technology (1999)
15. Mouchère, H., Viard-Gaudin, C., Zanibbi, R., Garain, U.: Icfhr2016 crohme: Competition on recognition of online handwritten mathematical expressions. In: 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR). pp. 607–612. IEEE (2016)
16. Mouchère, H., Viard-Gaudin, C., Kim, D.H., Kim, J.H., Garain, U.: Icfhr 2012 competition on recognition of on-line mathematical expressions (crohme 2012).

In: 2012 International Conference on Frontiers in Handwriting Recognition. pp. 811–816. IEEE (2012)
17. Mouchère, H., Viard-Gaudin, C., Zanibbi, R., Garain, U.: Icfhr 2014 competition on recognition of on-line handwritten mathematical expressions (crohme 2014). In: 2014 14th International Conference on Frontiers in Handwriting Recognition. pp. 791–796. IEEE (2014)
18. Mouchère, H., Viard-Gaudin, C., Zanibbi, R., Garain, U., Kim, D.H., Kim, J.H.: Icdar 2013 crohme: Third international competition on recognition of online handwritten mathematical expressions. In: 2013 12th International Conference on Document Analysis and Recognition. pp. 1428–1432. IEEE (2013)
19. Mouchère, H., Zanibbi, R., Garain, U., Viard-Gaudin, C.: Advancing the state of the art for handwritten math recognition: the crohme competitions, 2011–2014. International Journal on Document Analysis and Recognition (IJDAR) **19**, 173–189 (2016)
20. Oguiza, I.: tsai - a state-of-the-art deep learning library for time series and sequential data. Github (2023), https://github.com/timeseriesAI/tsai
21. Pastor, M., Toselli, A., Vidal, E.: Writing speed normalization for on-line handwritten text recognition. In: Eighth International Conference on Document Analysis and Recognition (ICDAR'05). pp. 1131–1135. IEEE (2005)
22. Peng, S., Gao, L., Yuan, K., Tang, Z.: Image to latex with graph neural network for mathematical formula recognition. In: Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II 16. pp. 648–663. Springer (2021)
23. Rahimian, E., Zabihi, S., Atashzar, S.F., Asif, A., Mohammadi, A.: Xceptiontime: independent time-window xceptiontime architecture for hand gesture classification. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1304–1308. IEEE (2020)
24. Rekers, J., Schürr, A.: Defining and parsing visual languages with layered graph grammars. Journal of Visual Languages & Computing **8**(1), 27–55 (1997)
25. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE transactions on neural networks **20**(1), 61–80 (2008)
26. Tang, J.M., Guo, H.Y., Wu, J.W., Yin, F., Huang, L.L.: Offline handwritten mathematical expression recognition with graph encoder and transformer decoder. Pattern Recognition p. 110155 (2023)
27. Tang, J.M., Wu, J.W., Yin, F., Huang, L.L.: Offline handwritten mathematical expression recognition via graph reasoning network. In: Pattern Recognition: 6th Asian Conference, ACPR 2021, Jeju Island, South Korea, November 9–12, 2021, Revised Selected Papers, Part I. pp. 17–31. Springer (2022)
28. Truong, T.N., Nguyen, H.T., Nguyen, C.T., Nakagawa, M.: Learning symbol relation tree for online mathematical expression recognition. arXiv preprint arXiv:2105.06084 (2021)
29. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
30. Wang, J., Du, J., Zhang, J., Wang, B., Ren, B.: Stroke constrained attention network for online handwritten mathematical expression recognition. Pattern Recognition **119**, 108047 (2021)
31. Wu, C., Du, J., Li, Y., Zhang, J., Yang, C., Ren, B., Hu, Y.: Tdv2: A novel tree-structured decoder for offline mathematical expression recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, pp. 2694–2702 (2022)

32. Wu, J.W., Yin, F., Zhang, Y.M., Zhang, X.Y., Liu, C.L.: Graph-to-graph: towards accurate and interpretable online handwritten mathematical expression recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 2925–2933 (2021)
33. Xie, Y., Mouchère, H., Simistira Liwicki, F., Rakesh, S., Saini, R., Nakagawa, M., Nguyen, C.T., Truong, T.N.: Icdar 2023 crohme: Competition on recognition of handwritten mathematical expressions. In: International Conference on Document Analysis and Recognition. pp. 553–565. Springer (2023)
34. Ye, J.Y., Zhang, Y.M., Yang, Q., Liu, C.L.: Contextual stroke classification in online handwritten documents with graph attention networks. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 993–998. IEEE (2019)
35. Yuan, Y., Liu, X., Dikubab, W., Liu, H., Ji, Z., Wu, Z., Bai, X.: Syntax-aware network for handwritten mathematical expression recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4553–4562 (2022)
36. Zhang, J., Du, J., Yang, Y., Song, Y.Z., Wei, S., Dai, L.: A tree-structured decoder for image-to-markup generation. In: International Conference on Machine Learning. pp. 11076–11085. PMLR (2020)
37. Zhang, T., Mouchère, H., Viard-Gaudin, C.: A tree-blstm-based recognition system for online handwritten mathematical expressions. Neural Computing and Applications **32**, 4689–4708 (2020)
38. Zhao, W., Gao, L., Yan, Z., Peng, S., Du, L., Zhang, Z.: Handwritten mathematical expression recognition with bidirectionally trained transformer. In: Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II 16. pp. 570–584. Springer (2021)