Long-Range Graph U-Nets: Node and Edge Clustering Pooling Model For Stroke Classification in Online Handwritten Documents

Muwu Yao[1*†], **Shuang She**[2*], **Jinrong Li**[2], **Jianmin Lin**[2‡], **Ming Yang**[2], **Hongxing Peng**[1‡]
[1] *College of Mathematics and Informatics, South China Agricultural University;* [2] *CVTE Research*

Editors: Berrin Yanıkoğlu and Wray Buntine

Abstract

Stroke classification is a crucial step for applications with online handwritten input. It is a challenging task due to the variations in writing style, complex structure, long contextual semantic dependence of written content and etc. In this work, we propose a method called Long-Range Graph U-Nets, which involves using a novel node and edge clustering graph pooling layer in the encoder block and a multi-level feature fusion strategy. Such operations guide the model to leverage both temporal and spatial contextual information, establish long-range semantic dependencies, and effectively reduce redundant information caused by local instances of the same category. Extensive experiments conducted on publicly available online handwritten document datasets, demonstrate that our proposed method outperforms previous methods by a significant margin, particularly in the List category, and achieves state-of-the-art performance.

Keywords: Stroke Classification, Graph Neural Networks, Clustering Graph Pooling

1. Introduction

With the broad application of intelligent interactive tablets, automatic recognition of freely handwritten ink documents has become an increasingly important research topic. Stroke classification, aiming to divide the strokes into different categories, is a key component of the handwritten ink analysis system. The difficulty of the problem can be summarized into two aspects, one is the complexity of shape structure, variation of writing styles and complex 2D structure. The other is the association of contextual semantics. Strokes of different categories may have similar shapes, while contextual semantics is the key to distinguishing them. As shown in Fig. 1, the main difference between the Text category and the List category lies in the numbering at the beginning.

Due to the inherent contextual relationships between strokes, such as spatial and temporal relationships, the problem of stroke classification is typically formulated using the structured prediction framework and tackled using various methods, including probabilistic graphical model (PGM), recurrent neural networks (RNN) and graph neural networks (GNN). The PGM-based methods (Ye et al., 2016) have disadvantages in computational

[‡] Equal contribution.
[‡] This work is done during MuwuYao's internship at CVTE Research.
[‡] Corresponding to linjianmin@cvte.com, xyphx@scau.edu.cn.

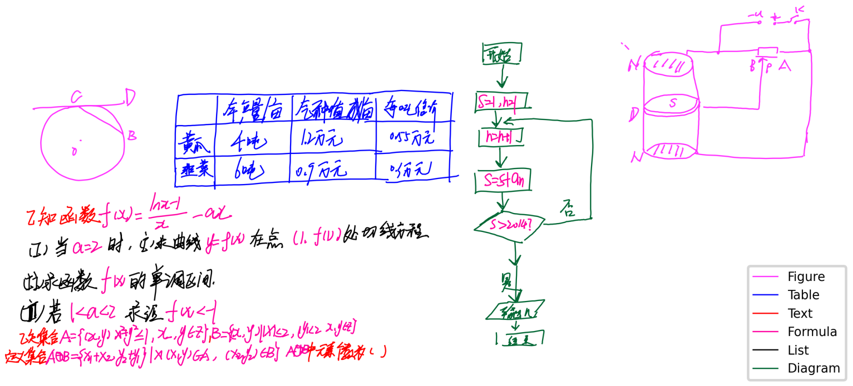


Figure 1: Example of online handwritten documents from CASIA-onDo.

complexity and model capacity. RNN-based methods (Grygoriev et al., 2021) have a strong capacity to model dependencies between temporally adjacent strokes, but they cannot make effective use of the spatial contextual relationships. Recently, GNN-based methods (Ye et al., 2019, 2020, 2021) have shown success in stroke classification problems, where each node represents a stroke and edges are constructed based on the temporal and spatial relationship between strokes. For multi-class classification, the current state-of-the-art(SOTA) method EGAT (Ye et al., 2020) achieves 89.87% on CASIA-onDo (Yang et al., 2021) dataset. However, as mentioned in (Yang et al., 2021), this model performs poorly in distinguishing between list and text, as well as inline formulas and text. We argue that **this model only aggregates information between neighborhood nodes, and these neighboring nodes usually belong to the same category**. This limitation results in restricted perception and the presence of redundant information in graph-level representations. As the semantic information of strokes is highly dependent on other key stroke information to form a complete semantic structure, addressing long-range dependencies and redundancy is crucial for accurate stroke classification.

In this paper, we propose a Long-Range Graph U-Nets with an encoder-decoder architecture to address above problems through two approaches: 1) a local node and edge clustering pooling strategy; 2) a multi-level feature fusion strategy. Pooling layers play a significant role in convolutional neural networks, largely due to their ability to reduce the sizes of feature maps and enlarge receptive fields, which leads to better generalization and performance. Many works (Liu et al., 2021; Itoh et al., 2022) have extended pooling operations to graphs successfully, but most of them focus on node pooling while ignoring edge information. In stroke classification, it is beneficial to have a module that allows edge information to communicate directly with adjacent edges to obtain long-range contextual semantic information. Therefore, we propose the node and edge clustering graph pooling (CPool) and un-pooling (UCPool) operations. This pooling mechanism condenses the input graph with node representations learned by GNN into a smaller-sized graph. However, stroke classification is a node-level task. Building on our proposed CPool and UCPool layers, we develop an encoder-decoder model on the graph, similar to U-Nets. To better capture the spatial and temporal information, we propose a multi-level feature fusion strategy to improve the model's performance.

The proposed method has several novelties and advantages:

• We propose a new clustering graph pooling/un-pooling strategy that fuses information from both nodes and edges simultaneously. This strategy preserves and propagates rich temporal-spatial context features to expand the receptive field and capture long-range dependencies.

• Building on these graph pooling/un-pooling operations and EGAT, we develop a Graph U-Nets architecture for stroke classification. Our proposed multi-level feature fusion strategy comprises two major components: 1) Node features are extracted by combining hand-crafted features with auto features extracted using BiLSTM. 2) We include skip connections between corresponding blocks of the encoder and decoder layers.

• We conducted extensive experiments on four popular public stroke datasets and achieved SOTA results. On the largest CASIA-onDo dataset, our model Long-Range Graph U-Nets (LR Graph) yields accuracy 93.56% for multi-class classification, and is a remarkable improvements especially on the class 'List'.

2. Related Work

2.1. Stroke Classification for Online Handwritten Documents

Exploiting temporal-spatial context for stroke classification is crucial. In the past, various PGM methods, such as the Hidden Markov Model (HMM) and Conditional Random Fields (CRF), have been used to exploit contextual relationships for stroke classification. HMM assumes strokes are independent of each other, leading to limited effectiveness for stroke classification. To overcome this drawback, (Ye et al., 2016) proposed a combined model of neural networks and CRF by training all parameters simultaneously. However, the PGM-based method has difficulty utilizing spatiotemporal contextual relationships effectively.

In contrast, RNNs can establish temporal contextual relationships effectively. (Grygoriev et al., 2021) proposed a hierarchical convolutional recurrent network (HCRNN), which uses hierarchical Bidirectional LSTM (BiLSTM) to learn the inherent hierarchical structure of the online handwritten document (sampling points-strokes or trajectory-document objects). However, due to the sequential structure of RNN, this type of network has difficulty utilizing spatial contextual information, leading to incomplete semantic contextual information during stroke classification and causing classification ambiguity.

GNNs have powerful modeling capabilities for complex contextual relationships. (Ye et al., 2019) proposed the GAT to establish the context of strokes. To address the issue of attention weights in GAT being obtained only through node features and ignoring edge features, (Ye et al., 2020) added the edge feature to GAT and designed the EGAT. However, EGAT heavily relies on hand-crafted feature extraction, which is problem-specific and requires professional experience. To simplify feature extraction, (Yang et al., 2021) propose adding a BiLSTM module before EGAT to extract node features. Although this approach simplifies feature extraction, it still lacks graph-level representations, suffers from limited receptive fields and redundant adjacent node information.

2.2. Graph Neural Networks

GNNs have found applications in various fields due to their ability to efficiently capture object and Non-Euclidean domain dependencies. (Kipf and Welling, 2016) proposed GCN, which introduces graph convolution kernels to construct an efficient and simple information transmission pathway that outperforms traditional GNN. (Veličković et al., 2017) proposed GAT, which add an attention mechanism to GCN to determine the weights of aggregated neighboring nodes. However, these models, such as GCN and GAT, only utilize the flat structure of the graph and do not consider graph-level representations.

Graph pooling layers can effectively learn graph-level representations, accelerates information exchange between nodes, and are roughly divided into flat pooling and hierarchical pooling. The former (Gao and Ji, 2019) directly generates a graph-level representation in one step, while the latter (Ying et al., 2018; Diehl, 2019) coarsens the graph gradually into a smaller sized graph. In (Gao and Ji, 2019), the top-k method is adopted to discard nodes with lower scores. Its node selection relies on global status and may discard nodes in the entire region, resulting in information loss. (Ying et al., 2018) proposed DiffPool based on clustering, which learns the feature of each node and soft-assigns each node to a fixed number of clusters to shrink the original graph. Cluster assignment based only on node feature may result in an unnecessary merge, ignoring distance information. (Diehl, 2019) further propose an edge-based associative progressive shrinkage clustering graph pooling method, which requires a amount of computation for single-edge contraction multiple times.

For the stroke classification problem, designing a model that can handle long-range semantic dependencies and reduce local redundant information of the same category is crucial. Graph pooling is an effective way to enlarge the receptive field and reduce redundancy. However, both dropout pooling and clustering pooling have their own disadvantages. Drop pooling can cause significant contextual information loss and orphaned nodes. Clustering pooling requires a high amount of computation. Therefore, designing efficient and effective pooling methods and integrating them into existing graph neural networks to achieve better performance in node-level tasks remains a key challenge.

3. Proposed Method

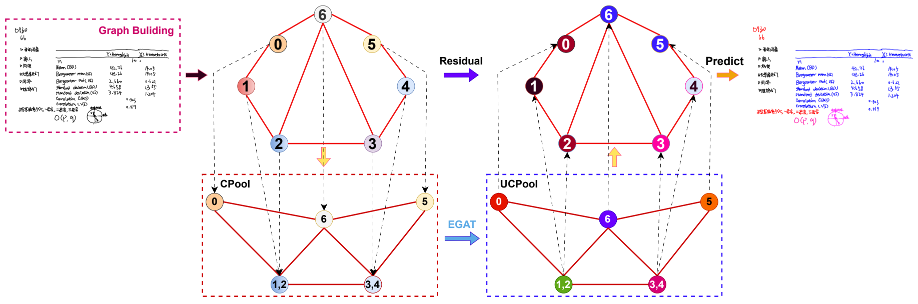


Figure 2: The overall architecture of our model. The node fusion uses dashed arrows, while the edge relationships are represented with solid lines.

An online handwritten document consists of a set of variable-length stroke sequences $S = \{s_i, i = 1...T\}$, where each stroke is comprised of a series of sample points. The objective is to train a model that can accurately classify the stroke sequences in the online document. Fig. 2 depicts the key components of our model, including graph building, cluster graph pooling layer, un-pooling layer as the inverse operation of the pooling layer, and the U-Nets-Like encoder-decoder architecture.

3.1. Graph Building

Graph Construction. We construct a relation graph $G = \{V, E\}$ based on the relationships between the strokes in an online document. Each stroke $s_i \in S$ is represented as a node $v_i \in V$, and the edge $e_{ij} \in E$ is defined as the contextual relationship between a pair of strokes (s_i, s_j). As demonstrated in (Ye et al., 2020), strokes that are temporally or spatially adjacent in an online document are often strongly correlated. To leverage both temporal and spatial contextual information, we consider two different edge construction approaches to build the edge set as referred in (Ye et al., 2020). One is the K-nearest temporal edges. For each stroke s_i, its forward k_t nearest strokes and backward k_t nearest strokes in the drawing sequence are considered as the temporal neighbors $N_T(s_i)$ of s_i. The other is K-nearest spatial edges. For each stroke s_i, its k_s nearest strokes in Euclidean space are identified as the spatial neighbors $N_S(s_i)$ of s_i. The distance between two strokes is defined as the minimal Euclidean distance among all pairwise points on the strokes. The hyperparameters k_t and k_s control the number of edges in the graph.

Feature Extraction. To account for different sampling rate on different hardware devices, we resample the origin points of each stroke s_i to k points based on equidistant resample and normalize the entire document to a resolution of 100x100. We then extract hand-crafted features (Indermühle et al., 2010) and perform feature normalization as their initial representations. For each node, a 13-dimensional vector of geometric features and a 10-dimensional vector of local context features are extracted from the corresponding stroke. For each edge, a 19-dimensional vector of features is extracted to model the spatial and temporal relationship between the two connected strokes. However, hand-crafted features may suffer from limited descriptive ability. To overcome this limitation, we also use an automatic stroke feature extraction method (Yang et al., 2021). For each node, the resampled points are processed into a 9-dimensional vector using first- and second-order derivatives. The encoded points are then input to a stroke feature extraction network based on BiLSTM. For each edge, the coordinate differences between the sampling points of two strokes are taken as the original edge feature.

Multi-Level Feature Fusion. The hand-crafted features characterize the information between strokes and stroke pairs based on different distance metrics, including temporal and spatial contextual features. In contrast, the automatically extracted node features focus more on the strokes themselves while ignoring their relationships with adjacent nodes. The automatic edge features are obtained by subtracting position information from adjacent strokes in space but do not consider temporal relationships between strokes.

We propose using the EGAT model to encode the hand-crafted features and concatenating them with the automatically extracted features to supplement the missing temporal and spatial contextual information in the automatic feature extraction process. The multi-level feature fusion also includes skip connections, which we will describe in the next section on how and why we use them.

3.2. CPool Layer and UCPool Layer

This section introduces the node and edge clustering graph pooling (CPool) layer and graph un-pooling (UCPool) layer. Fig. 3 provides a graphical illustration of these layers.

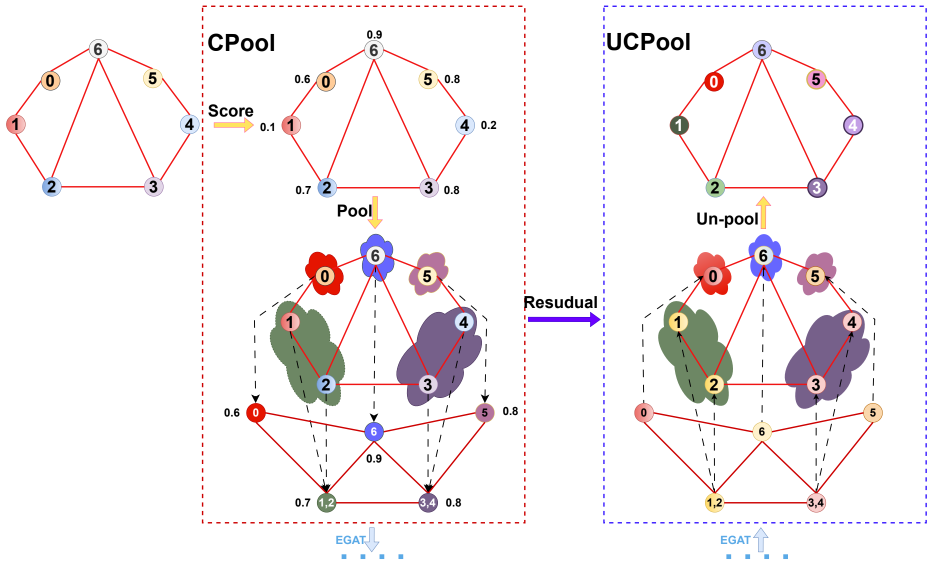


Figure 3: CPool and UCPool Layers.

CPool Layer. The clustering graph pooling (CPool) layer performs down-sampling on graph data, preserving graph-level representations by iteratively coarsening the graph into a smaller size. Clustering graph pooling considers graph pooling as a node and edge clustering problem, where nodes are mapped to a set of clusters that are treated as new nodes of the new graph. We adaptively select critical nodes for preservation while pooling the features of non-critical nodes and edges. If there is interaction between the nodes in two clusters, a new edge is created between them. Learning graph-level representations enhances the model's ability to capture long-range semantic dependencies.

Assume a graph $G = (V, E)$ with node features $H = \{h_i, i \in V\}$ where $h_i \in R^{D_1}$, and edge features $F = \{f_{ij}, (i, j) \in E\}$ where $f_{ij} \in R^{D_2}$. Here, D_1 and D_2 represent the dimensions of the node and edge features, respectively. V is the set of vertices, E is the set of edges, and the relationships between nodes can be represented by the adjacency matrix A. G^c represents the graph after pooling, which also contracts with node features H^c, edge features F^c, and adjacency matrix A^c. Specifically, node and edge clustering graph pooling can be deconstructed into three modules:

1) The Cluster Assignment Matrix (CAM) Construction. CAM is used to partition the nodes into clusters based on node and edge features. $CAM \in R^{n\times m}$ records the relationships between the clusters when n is the number of nodes in G and m is the number of nodes in G^c. We define the keep rate (kr) as the proportion of the original n nodes that are collapsed into m nodes, where $m = n \times kr$. A value of $C[x, y] = 1$ in CAM indicates that the x-th node is assigned to the y-th cluster.

$$C = CAM(H, F, A) \tag{1}$$

The core of CAM is the adaptive selection of more representative nodes as centers and the fusion of other nodes into representative nodes. The selection of important nodes not only depends on their geometric relationships but also on the influence of their edges. Therefore, we aggregate the edge information to comprehensively select the key nodes. We use two common pooling functions (maximum pooling and average pooling) to aggregate edge feature information connected to it onto nodes:

$$t_i^{max} = maxpool_{(i,j)\in E}\{f_{ij}\} \tag{2}$$

$$t_i^{ave} = avepool_{(i,j)\in E}\{f_{ij}\} \tag{3}$$

The edge feature is aggregated into its endpoint node v_i. Then, we merge the node features with the aggregated edge feature and project the resulting features using a trainable projection vector W_{score} to one dimension, forming a score for each node. Considering computational efficiency, we finally perform k-max pooling to select the top k nodes based on their scores.

$$r^{node} = \sigma(W_{score\_node} H) \tag{4}$$

$$r^{max} = \sigma(W_{score\_max} t^{max}) \tag{5}$$

$$r^{ave} = \sigma(W_{score\_ave} t^{ave}) \tag{6}$$

$$score = (W_{score}[r^{node}||r^{max}||r^{ave}])/||W_{score}|| \tag{7}$$

$$idx = rank(score, k) \tag{8}$$

$$Y = sigmoid(score) \tag{9}$$

Here, $W_{score\_node} \in R^{D_1' \times D_1}$ and $W_{score} \in R^{1\times(D_1'+2D_2')}$ are learnable parameter which maps feature to a new feature space. $W_{score\_max} \in R^{D_2' \times D_2}$ and $W_{score\_ave} \in R^{D_2' \times D_2}$ are learnable parameters that encode the maximum and average values of the edge features. $\sigma(\cdot)$ is leaky ReLU activation function, $|\cdot|$ represents concatenation, idx represents the central node in the new graph and Y represents the normalized scores of those nodes.

The correct classification of a stroke is strongly related to its adjacent nodes. Nodes with high scores need to aggregate the features of unselected nodes to preserve their strong contextual relationships. As demonstrated in (Ye et al., 2020), temporal information in online documents is critical. Thus, for nodes with low scores, we do not drop them and propose a simple method for selecting nodes to merge with them. Specifically, low-scoring nodes are merged with the nearest high-scoring nodes based on their temporal distance.

2) New Graph Generation. The structure of the new graph is obtained by the above clustering pooling from G. So the new adjacency matrix A^c represents the new graph relationships structured based on the A and CAM.

$$A^c = pool(A, C) = C^T AC \tag{10}$$

3) Feature Update. The features of the nodes and edges in the new graph are formed by aggregating from the old graph based on CAM. For node features in G^c, we obtain them using the following equations:

$$H^c = f_{node}(H, C) = C^T H \tag{11}$$

$$H^c = H^c \odot Y[idx] \tag{12}$$

where idx represents the central node in the new graph, and $Y[idx]$ represents the score of the node in the new graph, we assign different weights to H^c based on their scores.

To establish the long-range semantic dependencies in stroke classification, we need to design a reasonable method to generate edge features in G^c. It is beneficial to have a module that allows edge features to communicate directly with adjacent edges. We aggregate three different edge features. The first part of edge features is obtained based on the connected nodes.

$$r_{ij}^n = \sigma(W_{node}[h_i^{d_1}||h_j^{d_1}]) \tag{13}$$

Additionally, the other two parts of the edge features are mapped from G. We combine the three parts of edge features to generate the final edge features F_{ij}^c.

$$ct_i^{max} = C^T t_i^{max} \tag{14}$$

$$ct_i^{ave} = C^T t_i^{ave} \tag{15}$$

$$r_{ij}^{max} = \sigma(W_{max}[ct_i^{max}||ct_j^{max}]) \tag{16}$$

$$r_{ij}^{ave} = \sigma(W_{ave}[ct_i^{ave}||ct_j^{ave}]) \tag{17}$$

$$f_{ij}^c = \sigma(W_{edge}[r_{ij}^n||r_{ij}^{max}||r_{ij}^{ave}]) \tag{18}$$

To increase the stability of training, we add batch normalization to the node and edge updated features(H^c and F^c).

UCPool Layer. The core of UCPool is to restore the structure of G and embed the learned hierarchical features into it. Assuming G^u is the new graph with node features H^u, edge features F^u, and adjacency matrix A^u generated by applying UCPool to G^c. To implement the inverse operation of CPool, we need to re-establish A and CAM corresponding to the CPool Layer, which involves two independent modules:

1) Generate The Relationship of Graph G^u. The structure of G^u is the same as that of G, $A^u = A$. Therefor, the nodes and edges from same cluster have same features. How to enrich the features to enhance the model's representational power?

2) Generate The Features of G^u. We use a residual module that skip connections the features from G^c and G^u.

$$H^u = f_{node}^u(H^c, H, C) = CH^c + H \tag{19}$$

Referring to formulas (2) and (3), we merge the edge features of G^c to the nodes in G^c , get t_i^{maxc} and t_i^{avec} represented the edge feature of G^c and then restoring them to the nodes in G^u based on the CAM. As follow:

$$ct_i^{maxu} = Ct_i^{maxc} + t_i^{max} \tag{20}$$

$$ct_i^{aveu} = Ct_i^{avec} + t_i^{ave} \tag{21}$$

We obtained the node information $r_{ij}^{n\ u}$, edge maximum pooling r_{ij}^{maxu}, and average pooling r_{ij}^{aveu} of the endpoint nodes based on equations (13), (16), and (17), respectively. Finally, we incorporate the F into F^u.

$$r_{ij}^{e\ u} = \sigma(W_{edge}^{fu} f_{ij}) \tag{22}$$

$$f_{ij}^u = \sigma(W_{edge}^u [r_{ij}^{n\ u} || r_{ij}^{e\ u} || r_{ij}^{maxu} || r_{ij}^{aveu}]) + f_{ij} \tag{23}$$

where both $W_{edge}^{fu} \in R^{D_2' \times D_2}$ and $W_{edge}^u \in R^{D_2' \times 4D_2}$ are a learnable parameter.

3.3. Long-Range Graph U-Nets

Building on CPool and UCPool layers, we construct an encoder-decoder architecture called Long-Range Graph U-Nets. Each encoder layer is composed of a CPool and EGAT layer. The EGAT layer aggregates information from each node and edge, while CPool captures the long-range contextual semantic dependencies. In the decoder block, we stack the same number of decoding blocks as in the encoder part. Each decoder layer consists of a UCPool and EGAT layer. The UCPool layer restores the graph to its higher-resolution structure and fuses the features from the corresponding CPool layer through multi-level residuals. Fig. 4 provides a graphical illustration of this Graph U-Nets architecture. It is worth noting that there is an EGAT layer before each CPool layer and UCPool layer, allowing both the encoder and decoder to implicitly capture the topological information in the graph.

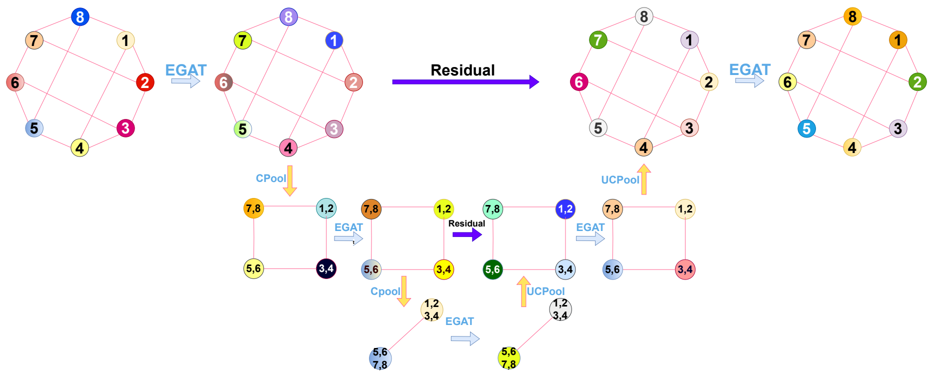


Figure 4: The framework of Long-Range Graph U-Nets.

4. Experiment

4.1. Datasets

CASIA-onDo (Yang et al., 2021) is the largest online handwritten document dataset for document layout analysis, which contains 2012 documents.

IAMonDo (Indermühle et al., 2010) is a publicly available online English handwritten document dataset, which contains 941 documents.

FC (Awal et al., 2011) is a publicly available online handwritten dataset of flowcharts, which contains 419 documents produced by 46 writers. As FC lacks a validation set, we followed the set paritioning by (Ye et al., 2020).

FA (Bresler et al., 2014) is a finite-state automaton dataset, which contains 300 diagrams produced by 25 writers.

4.2. Training

Hyper-Parameters. For stroke feature extraction, we set the number of sampling points k to 20, and we use a BiLSTM module with 50 hidden units consisting of 3 layers. Additionally, we set the attention heads (H) in EGAT layer to 8. kr is set to 0.7 and dropout rate is set to 0.2. We conduct the experiments on a single NVIDIA 1080Ti GPU, training the network for 300 epochs using the AdamW optimizer with a cosine annealing learning rate optimization algorithm. Other hyper-parameters, such as k_t, number of node output features (D'_1), number of edge output features (D'_2), and pooling layers(L) are adjusted through random search on the validation set. Details are provided in Table 1.

Table 1: Hyper-parameters for all experiments

Hyperparameters	CASIA-onDo-2	CASIA-onDo-6	IAMonDo-2	IAMonDo-5	FC	FA
k_t	8	8	2	2	1	1
k_s	8	8	6	6	4	4
D'_1	32	32	32	32	32	28
D'_2	40	40	40	40	40	32
L	3	5	3	3	3	1
Batch size	13	11	18	18	40	40
Learning rate	0.003	0.0015	0.007	0.007	0.008	0.008

Evaluation Metrics. We evaluate our model on both text/non-text classification and multi-class classification tasks. For text/non-text classification, the accuracy used as the evaluation metric is defined as:

$$Accuracy = \frac{\sum_{i=1}^{N}\sum_{i=1}^{T_i}\delta(y_{it} = \hat{y_{it}})}{\sum_{i=1}^{N} T_i} \tag{24}$$

where N is the number of documents, and T_i is the number of strokes in the i-th document. $\hat{y_{it}}$ and y_{it} are the corresponding prediction and ground truth. For multi-class classification, the accuracy for each class c is defined as:

$$Accuracy[c] = \frac{\sum_{i=1}^{N}\sum_{i=1}^{T_i}\delta(y_{it} = c)\delta(\hat{y_{it}} = y_{it})}{\sum_{i=1}^{N}\sum_{i=1}^{T_i}\sigma(y_{it} = c)} \tag{25}$$

4.3. Experiments and Analysis

Text/Nontext Classification. We report the results of different methods on text/non-text classification tasks on the CASIA-onDo and IAMonDo in Tables 2 and 3, respectively. Our model achieves an accuracy of 97.68%(+0.96%) on the CASIA-onDo. On the IAMonDo, our model achieves an accuracy of 98.78%, which is 1.77% higher than the BiLSTM that only considers temporal context and 1.57% higher than the CRF with multiple context. Those indicate that our model better utilizes temporal and spatial contexts. Compared to EGAT, our model achieves a 1.56% and 0.13% improvement on the CAISA-onDo and IMAonDo datasets, respectively.

Table 2: Text/Nontext stroke classification on the CASIA-onDo

Method	Description	Accuracy
EGAT (Yang et al., 2021)	EGAT with hand-crafted feature.	96.12
EGAT(Auto) (Yang et al., 2021)	EGAT with BiLSTM.	96.72
LR Graph(ours)	Long-Range Graph U-Nets.	**97.68**

Table 3: Text/Nontext stroke classification on the IAMOnDo

Method	Description	Accuracy
(Indermuhle et al., 2012)	BLSTM network	97.01
(Delaye and Liu, 2014a)	CRF with multiple contexts	97.21
GCN (Ye et al., 2020)	Graph convolutional networks	97.21
GAT (Ye et al., 2020)	Graph attention networks	97.87
EGAT (Ye et al., 2020)	EGAT with hand-crafted feature.	98.65
LR Graph(ours)	Long-Range Graph U-Nets.	**98.78**

Multi-Class Classification. Tables 4, 5, 6, and 7 compare the results of our model for multi-classification on the CASIA-onDo, IAMonDo, FC, and FA. Our LR Graph achieves SOTA. On the CASIA-onDo, our model achieves the accuracy 93.56%, which is 3.69% than the EGAT(Auto). Except for the Formula, all other classes have an improvement, especially the List, which increased by 21.5%. The confusion matrix of the LR Graph on the CASIA-onDo is shown in Fig. 5, where the Text, Formula, and List categories demonstrate remarkable improvements. Similar conclusions are also demonstrated on the IAMonDo and an accuracy of 97.89% is achieved, representing an improvement of 2.08% over EGAT. Especially, the accuracy of the List increase by 22.65%. On the FC dataset, the accuracy of our model is 97.94% and is 0.58% higher than that of the previous best method. On the FA, our model yields 99.49% accuracy, which is 0.44% higher than that of the previous best method. Those results indicate that our model captures long-range dependency information and reduced redundant information through clustering graph pooling.

Table 4: Multi-Class stroke classification on the CASIA-onDo

Method	Text	Formula	Diagram	Table	Figure	List	Accuracy
EGAT (Yang et al., 2021)	89.59	76.94	98.17	94.50	91.26	76.13	88.79
EGAT(Auto) (Yang et al., 2021)	93.04	89.43	96.07	94.35	91.17	68.55	89.87
LR Graph(ours)	94.26	86.06	98.14	96.71	91.41	90.05	**93.56**

Table 5: Multi-Class stroke classification on the IAMonDo

Method	Graphics	Text	Table	List	Math	Accuracy
Loopy CRF (Delaye and Liu (2014b))						79.22
(Delaye and Liu, 2014b)	95.85	97.25	77.64	74.73	84.28	93.46
GCN (Ye et al., 2020)	93.09	97.97	70.31	52.73	74.48	91.11
GAT (Ye et al., 2020)	95.12	98.19	77.59	69.66	82.22	93.51
EGAT (Ye et al., 2020)	97.11	98.35	89.70	76.15	88.43	95.81
LR Graph(ours)	98.00	98.83	97.12	98.80	85.65	**97.89**

Table 6: Multi-Class stroke classification on the FC

Method	Text	Arrow	Data	Decision	Process	Terminator	Connection	Accuracy
(Wang et al., 2017)	99.0	92.1	84.4	89.9	93.5	78.9	79.3	95.8
(Bresler et al., 2016)	99.2	87.5	95.3	88.2	96.3	90.7	94.1	96.3
GCN (Ye et al., 2020)	97.74	89.44	88.67	92.92	82.30	85.96	85.19	93.99
GAT (Ye et al., 2020)	98.01	88.21	88.57	92.27	82.39	87.09	87.56	94.00
EGAT (Ye et al., 2020)	98.84	96.39	94.40	95.19	92.94	93.23	92.89	97.36
LR Graph(ours)	99.39	96.77	94.90	95.82	96.00	91.69	86.57	**97.94**

Table 7: Multi-Class stroke classification on the FA

Method	Label	Arrow	Initial arrow	State	Final state	Accuracy
(Wang et al., 2017)	99.0	97.7		91.6	96.5	97.8
(Bresler et al., 2016)	99.7	98.0	98.6	98.3	99.2	99.0
GCN (Ye et al., 2020)	94.04	91.75		84.18	87.81	92.13
GAT (Ye et al., 2020)	99.01	97.89		92.30	94.81	97.87
EGAT (Ye et al., 2020)	99.49	99.08		97.18	97.42	99.05
LR Graph(ours)	99.76	99.93		96.52	98.08	**99.49**

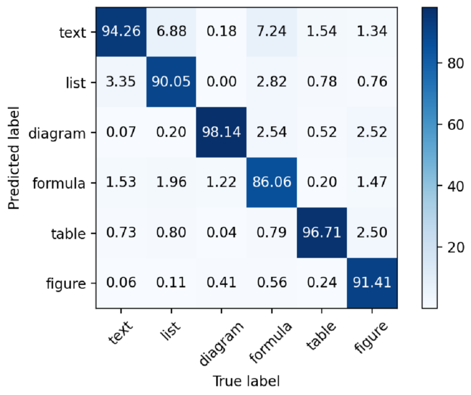


Figure 5: The confusion matrix of LR Graph on the he CASIA-onDo

Ablation Study. To determine the key factors of our proposed model, we conducted ablation studies in four different aspects based on the baseline EGAT. Table 8 shows the methods and their related modules.

Table 8: Methods and its associated modules. 'Hand' denotes hand-crafted features, 'Auto' denotes BiLSTM-extracted features, 'CPool and UCPool' denotes clustering pooling layers, and 'Multi-Residual' denotes skip connections from CPool to UCPool.

Method	Hand	Auto	CPool and UCPool	Multi-Residual
EGAT	✓			
EGAT(Auto)		✓		
LR Graph(Hand and Auto)	✓	✓		
LR Graph(Multi-Residual)	✓	✓		✓
LR Graph(Pool)	✓	✓	✓	
LR Graph	✓	✓	✓	✓

For combining hand-crafted features with auto features, LR Graph(Hand and Auto) gets accuracies of 91.40% on CASIA-onDo in Table 9. The result demonstrates that the concatenation of hand-crafted features and BiLSTM-extracted features can improve performance, especially for BiLSTM-extracted features, which focus more on spatial context but lack temporal context information, hand-crafted features can provide complete contextual information and enhance the model's performance. For Multi-Residual, LR Graph (Multi-Residual) outperformed LR Graph(Hand and Auto) by 0.72% in Table 9, LR Graph with an accuracy of 93.56% outperforms LR Graph(Pool) which represents an improvement of 1.48%. These results indicate that multi-level feature fusion enhance the model's expressive power by combining hand-crafted features with auto features and skip connections between shallow-level graph features and deep-level graph features.

Table 9: Ablation study on multi-class stroke classification on the CASIA-onDo

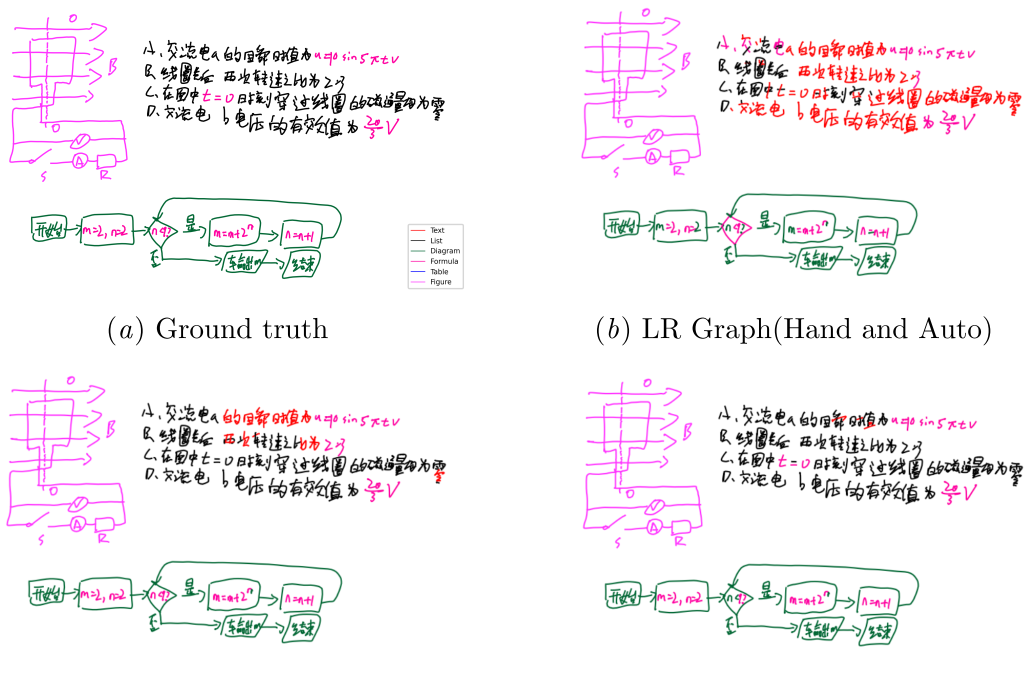
Method	Text	Formula	Diagram	Table	Figure	List	Accuracy
EGAT(Auto)	93.04	89.43	96.07	94.35	91.17	68.55	89.87
LR Graph(Hand and Auto)	92.90	81.66	97.24	95.85	91.91	85.50	91.40
LR Graph(Pool)	93.37	83.83	97.67	94.72	91.10	87.60	92.08
LR Graph(Multi-Residual)	94.11	84.00	97.65	95.36	90.89	84.70	92.12
LR Graph	94.26	86.06	98.14	96.71	91.41	90.05	93.56

Table 10: LR Graph(w, w/o cluster) on multi-class classification on the CASIA-onDo

Method	Text	Formula	Diagram	Table	Figure	List	Accuracy
LR Graph(w/o cluster)	93.00	82.97	98.11	95.85	90.00	89.97	92.44
LR Graph	94.26	86.06	98.14	96.71	91.41	90.05	93.56

Evaluation of CPool and UCPool. In Tables 9 and 10, we study the contribution of CPool and UCPool layers in LR Graph. We direct discard of nodes in pooling without cluster to obtain LR Graph(w/o cluster). LR Graph(w/o cluster) and LR Graph achieve 92.44% and 93.56%, respectively, with an improvement of 0.32% and 1.44% compared to LR Graph(Multi-Residual). Notably, LR Graph(Pool) has slightly lower accuracy than LR Graph (Multi-Residual), with a decrease of 0.04% but it can be seen that LR Graph(Pool) has a higher recall rate than LR Graph(Multi-Residual) in the List category, with an improvement of 2.9%. In the List category that requires long-range dependencies, LR Graph (w/o cluster) and LR Graph both achieve a significant improvement with an increase of 5.27% and 5.35%, respectively. This indicates that the model can learn graph-level representations and capture long-range dependencies under the pooling, which is a key feature of the LR Graph.

Visualization Analysis. With the color of the instance category referencing in Fig. 6(a), Fig. 6 shows the recognition results of LR Graph, LR Graph(Hand and Auto), and LR Graph(w/o cluster) on the same document. LR Graph and LR Graph(w/o cluster) effectively address the problem of long-range dependencies, as demonstrated in the classification of List compared with LR Graph(Hand and Auto) in Fig. 6(b). However, the loss of essential stroke information leads to the misclassification of most of the subsequent strokes in Fig. 6(c). Thus, LR Graph's ability to aggregate low-scoring nodes while maintaining high contextual information greatly improves performance, as evident in Fig. 6(d).



(*a*) Ground truth (*b*) LR Graph(Hand and Auto)

(*c*) LR Graph (w/o cluster) (*d*) LR Graph

Figure 6: The recognition results of different methods on CASIA-onDo

5. Conclusion

In this paper, we present a novel framework for stroke classification in online handwritten documents. Our approach employs a novel node and edge clustering graph pooling and un-pooling operations, which enlarge the model's receptive fields, enabling it to capture long-range dependencies and reduce redundant information. To improve generalization and performance, a multi-level feature fusion strategy is also used. Experiments conducted on four publicly available online handwritten document datasets, including CASIA-onDo, IAMonDo, FC, and FA, demonstrate that our proposed method outperforms previous methods by a significant margin, particularly in the List category, achieving state-of-the-art performance. While these results are encouraging, some challenges remain to be further explored, such as in our clustering graph pooling layer the low-scoring nodes are merged with the nearest high-scoring nodes based on their temporal distance, which may be replaced by a better mechanism and precise distinction between text and inline formula.

References

Ahmad-Montaser Awal, Guihuan Feng, Harold Mouchere, and Christian Viard-Gaudin. First experiments on a new online handwritten flowchart database. In *Document Recognition and Retrieval XVIII*, volume 7874, pages 81–90. SPIE, 2011.

Martin Bresler, Truyen Van Phan, Daniel Prusa, Masaki Nakagawa, and Vaclav Hlavac. Recognition system for on-line sketched diagrams. In *2014 14th International Conference on Frontiers in Handwriting Recognition*, Sep 2014.

Martin Bresler, Daniel Prša, and Václav Hlaváč. Online recognition of sketched arrow-connected diagrams. *International Journal on Document Analysis and Recognition (IJDAR)*, 19:253–267, 2016.

Adrien Delaye and Cheng Lin Liu. Contextual text/non-text stroke classification in online handwritten notes with conditional random fields. *Pattern Recognition*, 47(3):959–968, 2014a.

Adrien Delaye and Cheng-Lin Liu. Multi-class segmentation of free-form online documents with tree conditional random fields. *International Journal on Document Analysis and Recognition (IJDAR)*, 17:313–329, 2014b.

Frederik Diehl. Edge contraction pooling for graph neural networks. *arXiv preprint arXiv:1905.10990*, 2019.

Hongyang Gao and Shuiwang Ji. Graph u-nets. In *international conference on machine learning*, pages 2083–2092. PMLR, 2019.

Andrii Grygoriev, Illya Degtyarenko, Ivan Deriuga, Serhii Polotskyi, Volodymyr Melnyk, Dmytro Zakharchuk, and Olga Radyvonenko. Hcrnn: a novel architecture for fast online handwritten stroke classification. In *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II*, pages 193–208. Springer, 2021.

E Indermuhle, Volkmar Frinken, and Horst Bunke. *Mode Detection in Online Handwritten Documents Using BLSTM Neural Networks.* Mode Detection in Online Handwritten Documents Using BLSTM Neural Networks, 2012.

Emanuel Indermühle, Marcus Liwicki, and Horst Bunke. Iamondo-database: an online handwritten document database with non-uniform contents. In *Proceedings of the 9th IAPR international workshop on document analysis systems*, pages 97–104, 2010.

Takeshi D Itoh, Takatomi Kubo, and Kazushi Ikeda. Multi-level attention pooling for graph neural networks: Unifying graph representations with multiple localities. *Neural Networks*, 145:356–373, 2022.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. 2016.

Ning Liu, Songlei Jian, Dongsheng Li, Yiming Zhang, Zhiquan Lai, and Hongzuo Xu. Hierarchical adaptive pooling by capturing high-order dependency for graph representation learning. 2021.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Chengcheng Wang, Harold Mouchere, Christian Viard-Gaudin, and Lianwen Jin. Combined segmentation and recognition of online handwritten diagrams with high order markov random field. In *International Conference on Frontiers in Handwriting Recognition*, 2017.

Yu-Ting Yang, Yan-Ming Zhang, Xiao-Long Yun, Fei Yin, and Cheng-Lin Liu. Casia-ondo: a new database for online handwritten document analysis. In *Asian Conference on Pattern Recognition*, pages 174–188. Springer, 2021.

Jun-Yu Ye, Yan-Ming Zhang, and Cheng-Lin Liu. Joint training of conditional random fields and neural networks for stroke classification in online handwritten documents. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 3264–3269. IEEE, 2016.

Jun Yu Ye, Yan Ming Zhang, Qing Yang, and Cheng Lin Liu. Contextual stroke classification in online handwritten documents with graph attention networks. *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 2019.

Jun-Yu Ye, Yan-Ming Zhang, Qing Yang, and Cheng-Lin Liu. Contextual stroke classification in online handwritten documents with edge graph attention networks. *SN Computer Science*, 1:1–13, 2020.

Jun-Yu Ye, Yan-Ming Zhang, Qing Yang, and Cheng-Lin Liu. Joint stroke classification and text line grouping in online handwritten documents with edge pooling attention networks. *Pattern Recognition*, 114:107859, 2021.

Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31, 2018.