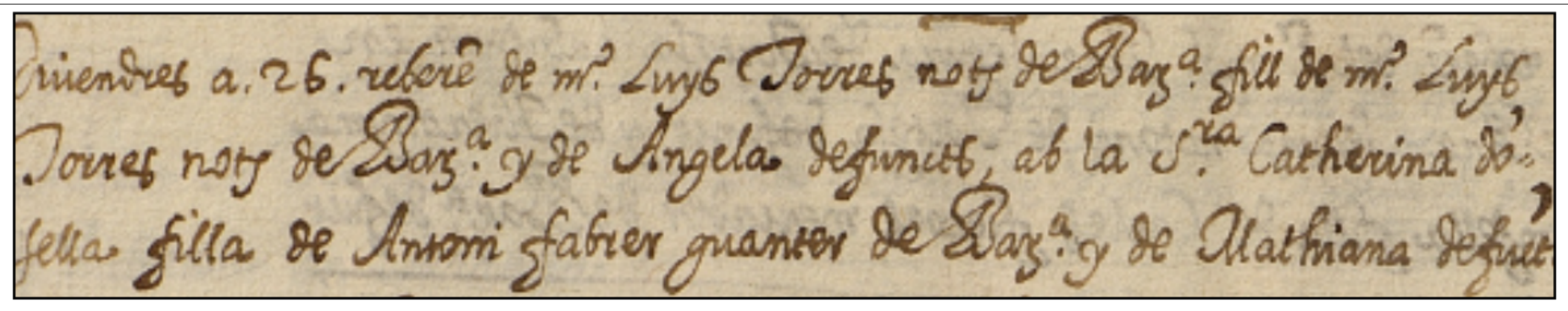
Graph Neural Networks for end-to-end information extraction from handwritten documents

Yessine Khanfir [1] Marwa Dhiaf [1,2] Ahmed Cheikhrouhou [1] Emna Ghodhbani [1] Yousri Kassentini [2]

[1]InstaDeep [2]Digital Research Center of Sfax

Problem formulation

Paper documents exist in different forms and hold valuable information. Historical records, for instance, may be used to determine ethnic origins or even to glean important historical information. Business and administrative documents, in turn, can be used to carry out statistical analyses. However, the large volume of data makes manual transformation impractical. Therefore, the adoption of **automated Information Extraction (IE) systems** is necessary to process these documents.



Divendres a 26 rebere de mr **[name_husband]** Luys **[surname_husband]** Torres **[occupation_husband]** nots de **[location_husband]** Bara fill de mr **[name_husbands_father]** Luys **[surname_husbands_father]** Torres **[occupation_husbands_father]** nots de **[location_husbands_father]** Bara y de **[name_husbands_mother]** Angela defuncts ab la Sra **[name_wife]** Catherina **[state_wife]** dosella filla de **[name_wifes_father]** Antoni **[surname_wifes_father]** fabrer **[occupation_wifes_father]** guanter de **[location_wifes_father]** Bara y de **[name_wifes_mother]** Mathiana defucts

Figure 1. Example of joint text and named entity recognition from a historical handwritten document

Automating IE from handwritten documents is a challenging task due to the wide variety of handwriting styles, the presence of noise, and the lack of labeled data. Information extraction approaches from document images are either based on a **two-stage** or an **end-to-end architecture**. For example, to perform **Named Entity Recognition (NER)**, a two-stage approach transforms the document image into a **textual representation**, and then, Natural Language Processing (NLP) techniques are applied to parse the output text and **extract the named entity tags**. On the other hand, the end-to-end method, also known as the **joint learning approach**, involves the **simultaneous recognition of text and Named Entity (NE) annotations**, or the direct **identification of NEs** on the image level without requiring an **explicit recognition step** at the text level.

Overview of the proposed approach

We propose an **end-to-end encoder-decoder model**, that combines transformers and Graph Convolutional Networks (GCN), to **jointly** handwritten text and named entity recognition. We introduce a **Sparse Graph Transformer Encoder (SGTE)** and a **Cross-GCN Deocder** to simultaneously take advantage of **self-attention mechanism** and **Graph Neural Networks (GNN)** in **representation learning** and **relation extraction**. Furthermore, our method benefits from the flexibility of graph structures to **control the scope of information propagation**.

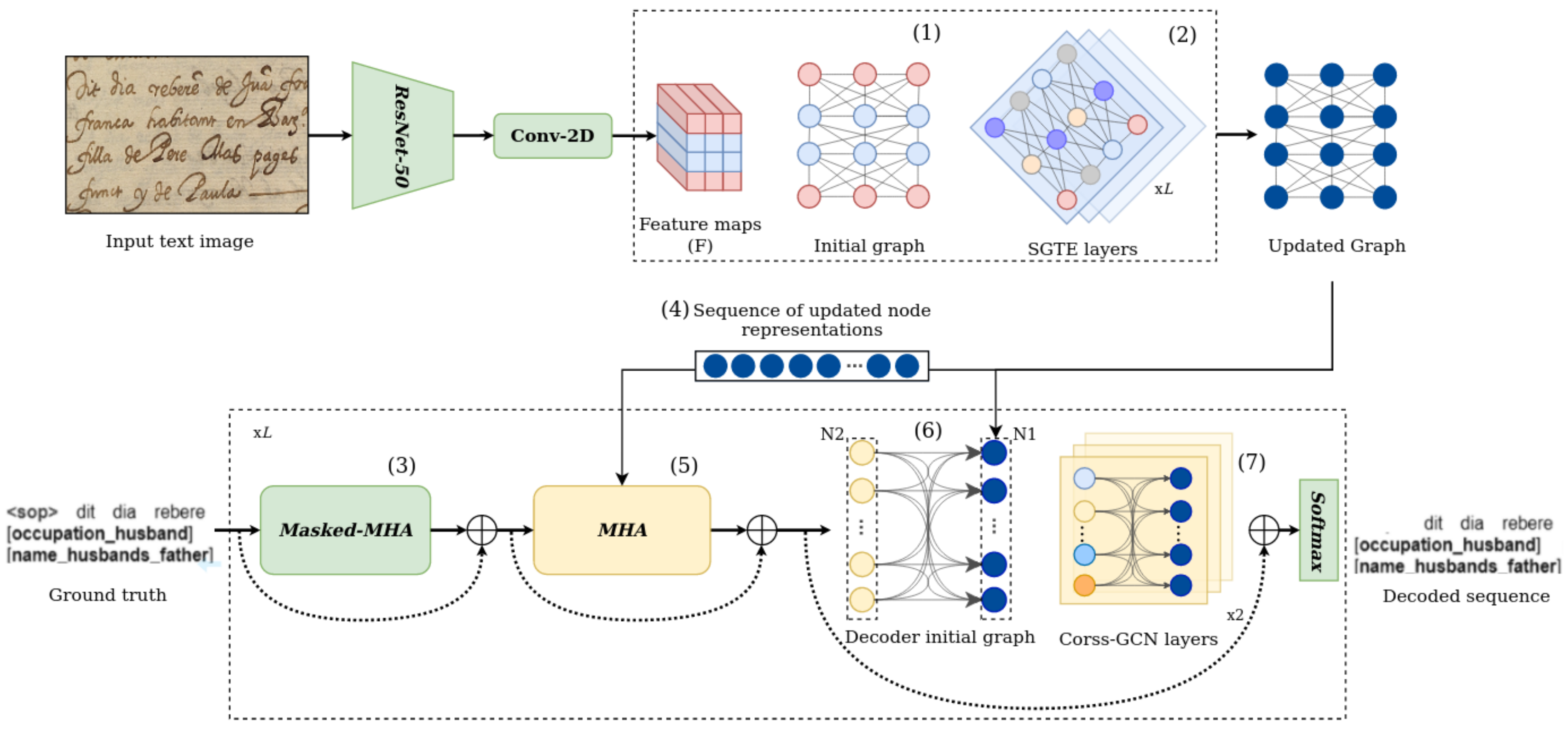


Figure 2. Overview of the proposed approach

The encoder-decoder form is preserved, as it is suitable for our **sequence-to-sequence learning task**. Input images are fed into a **pre-trained ResNet-50 for feature extraction**, followed by a 2D-convolutional layer with a kernel size of 1×1 to match the number of features from the backbone network and the encoder input. For the encoding part, we adopt **the generalization of transformers to graph structures** in Dwivedi et al [1].
The decoder of the traditional transformer model includes a **Masked Multi-Head Attention (MMHA) block** to **model relationships within the ground truth**, and a **Multi-Head Attention (MHA) block** responsible for the **alignment of the visual features to characters and NEs** through self-attention. In our proposed model, we extend this specific operation of alignment using the Cross-GCN in the decoder part, built from the output of the SGTE and the decoder MHA block.

Cost function

The SGTE and the Cross-GCN-based Decoder are **jointly trained and supervised** with the **categorical cross-entropy loss** computed based on the **sequence predicted by the Decoder and the target sequence**. We denote $\hat{Y} = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_T)$ as the predicted sequence and $Y = (y_1, y_2, ..., y_T)$ as the target sequence, where T represents the sequence length. The loss function L adopted to train our model can be formulated as follows:

$$L\left(\hat{Y}, Y\right) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{C} y_{t,i} log\left(p_{t,i}\right) \tag{1}$$

where,

$$p_t = softmax(\hat{y}_t) \tag{2}$$

Here C refers to the number of unique words in the vocabulary. $y_t\left(y_{t,1}, y_{t,2}, ..., y_{t,C}\right)$ denotes the one-hot encoded vector where $y_{t,i}$ is equal to 1 if it corresponds to the true class at step t, and 0 otherwise. $p_{t,i}$ corresponds to the predicted probability of the i_{th} word in the vocabulary at step t. The predicted probability distribution $p_t\left(p_{t,1}, p_{t,2}, ..., p_{t,C}\right)$ is obtained by applying a $softmax$ scaling to the model's predicted logits over the vocabulary $\hat{y}_t$.

Sparse Graph Transformer Encoder

Knowing that the **Graph Transformer** [1] operates on graphs, using it as an encoder requires a prior **graph construction** step. Figure 3 provides a realistic illustration of how the **initial graph** is constructed. Feature maps (F) resulting from the last convolution block are used to initialize the graph nodes. Indeed, elements that share the same spatial position in all feature maps are stacked and assigned as a **node representation** in an initial graph. For the sake of complexity reduction, we leverage **graph sparsity**, which also allows to **customize the scope of attention** of each element. Our strategy is to select each node's neighborhood according to its original position in F. As illustrated in step 1 of Figure 2, elements on the first line of F, are connected to all neighbors from the same and the next line. Elements on the last line of F, are connected to all neighbors from the same and the previous line. The remaining feature vectors are connected to the elements on the same, previous, and following line.

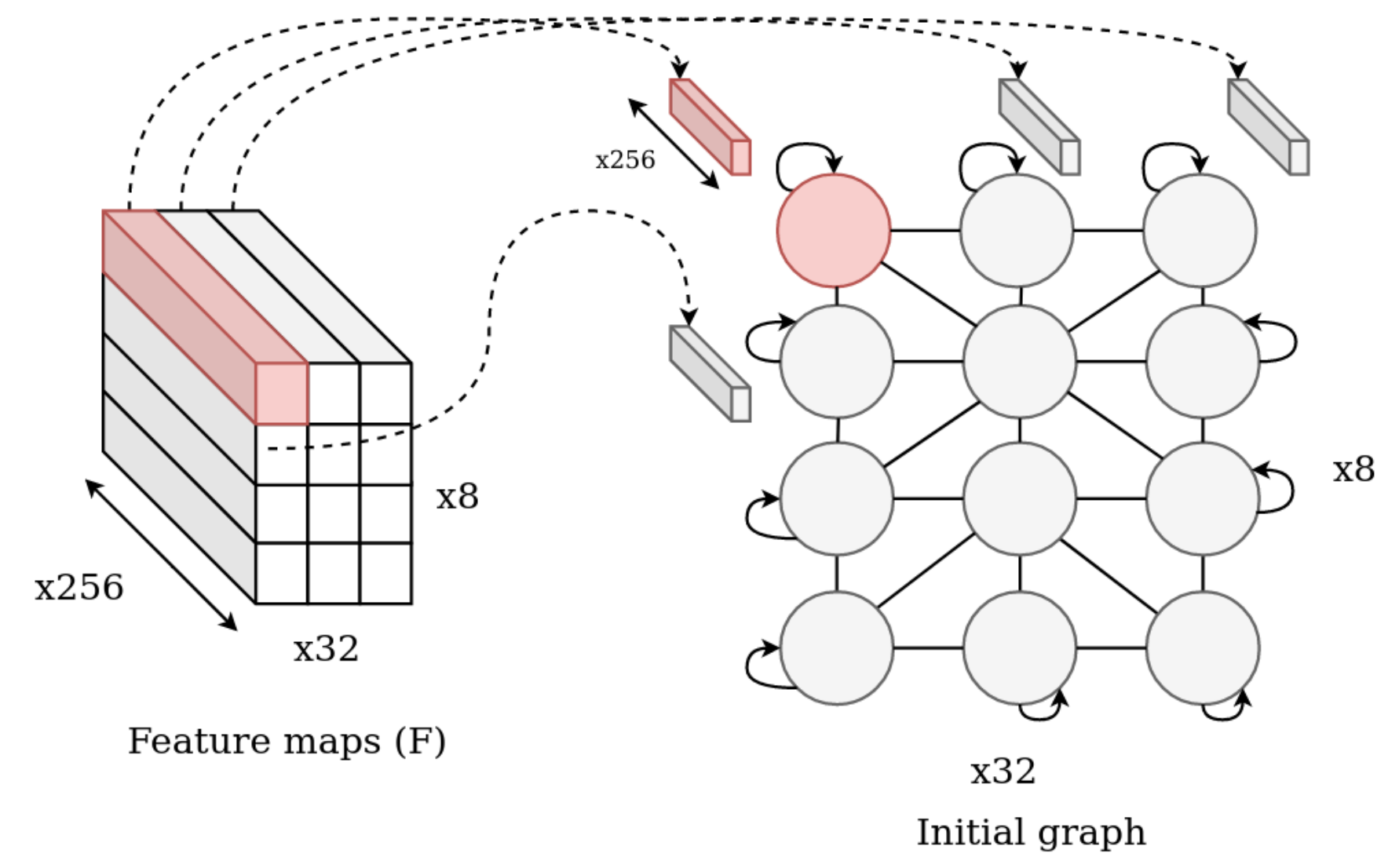


Figure 3. Initial encoder graph construction method

Mathematical formulation of SGTE

After the graph is built, it goes through the SGTE layers to update the node representations over the attention heads. For each layer l of the SGTE, the representation h_i^l of the i_{th} node is updated as follows:

$$\hat{h}_i^{l+1} = O_h^l \,\|_{k=1}^{H} \big(\sum_{j \epsilon N_i} w_{ij}^{k,l} V^{k,l} h_j^l\big), \tag{3}$$

where,

$$w_{ij}^{k,l} = softmax_j\Big(\frac{Q^{k,l}h_i^l \cdot K^{k,l}h_j^l}{\sqrt{d_k}}\Big), \tag{4}$$

and $Q^{k,l}, K^{k,l}, V^{k,l} \in R^{d_k * d}, O_h^l \in R^{d*d}, k \in [1, H]$ denotes the number of attention heads, $\|$ denotes concatenation. N_i refers to the set of nodes directly connected with edges to the i^{th} node.

Cross-GCN Decoder

In this part, we explore the effect of **reinforcing the representation learning** during the decoding step, with a **two layers GCN**, as detailed in step (7) of Figure 2. The goal is to jointly benefit from the **attention mechanism**, and **message-passing principles of graph convolutions** for a more robust **alignment of visual features to characters and NE tags**. To this end, as shown in step (6) of Figure 2, we construct a **directed graph** of nodes emerging from the output of the last SGTE layer (N1), and nodes emerging from the output of the MHA block (N2). In order not to cancel the **masking effect of the MMHA block**, N2 nodes are not connected to each other. Afterward, the decoder's initial graph is then fed to the Cross-GCN.

Mathematical formulation of Cross-GCN

In this block, each node coming from the MHA component will be represented, by the weighted sum over N_1 nodes. For each layer l of the Cross-GCN, the update of the representation h_i^l of the i_{th} node is computed as follows:

$$\hat{h}_i^{l+1} = \sigma\left(w \cdot h_i^l + W \cdot \sum_{j \epsilon N_1} \frac{h_j^l}{\sqrt{N_{(i)} + N_{(j)}}}\right) \tag{5}$$

Where σ is a non-linearity, w is a weight coefficient multiplied with the initial representation of the i_{th} node before aggregation, N_1 denotes the set of nodes originating from the last SGTE layer, W is a weight matrix, $N_{(i)}$ and $N_{(j)}$ refer to the degrees of the i_{th}, j_{th} nodes respectively, h_j^l is the representation l of the j_{th} node and $\frac{1}{\sqrt{N_{(i)}+N_{(j)}}}$ is added as a regularization term.

Results on handwritten text datasets

System	Basic	Complete	Level
Hitsz-ICRC-2	94.16	91.97	Word
CITLab-Argus-2	91.93	91.56	Line
Carbonell et al.	90.58	89.39	Line
Transformer [2]	95.16	93.3	Line
Transformer [2]	96.25	95.54	Record
Ours	**96.22**	**96.24**	Record

Table 1. Results on the Esposalles dataset

System	Precision	Recall	F1-score
Rowtula et al.	58.8	41.3	47.4
HTR-NER	77.3	65.9	70.7
HTR-D-NER	78.6	73.0	75.4
Annotation-NER [3]	83.8	77.5	80.1
Transformer [2]	98.1	71.4	82.6
Ours	**98.2**	**76.1**	**85.7**

Table 2. Results on the IAM dataset

References

[1] Vijay Prakash Dwivedi and Xavier Bresson.
A generalization of transformer networks to graphs.
AAAI 2021 Workshop on Deep Learning on Graphs: Methods and Applications, abs/2012.09699, 2020.

[2] Ahmed Cheikh Rouhou, Marwa Dhiaf, Yousri Kessentini, and Sinda Ben Salem.
Transformer-based approach for joint handwriting and named entity recognition in historical document.
Pattern Recognition Letters, 155:128–134, 2022.

[3] Oliver Tüselmann, Fabian Wolf, and Gernot A Fink.
Are end-to-end systems really necessary for ner on handwritten document images?
In *International Conference on Document Analysis and Recognition*, pages 808–822. Springer, 2021.