PICK: Processing Key Information Extraction from Documents using Improved Graph Learning-Convolutional Networks

Wenwen Yu*[1], Ning Lu*[2], Xianbiao Qi[2], Ping Gong[1], Rong Xiao[2]

[1]*Xuzhou Medical University, Xuzhou, China*
[2]*Ping An Property & Casualty Insurance Company, Shenzhen, China*

ICPR 2020
25th INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION
Milan, Italy 10 | 15 January 2021

Task Definition

- Key Information Extraction (KIE) from documents is the downstream task of OCR.
- The aim of KIE is to extract a number of key fields from the given documents, and save the texts to structured documents.
- KIE is essential for a wide range of technologies such as efficient archiving, fast indexing, document analysis and so on.

Motivation

- KIE is a challenge task because documents not only have textual features extracting from OCR systems, but also have semantic visual features that are not fully exploited, and it play a critical role in KIE.
- Too little work has been devoted to efficiently make full use of both textual and visual features of the documents.
- Existing methods for KIE only use text and box, and need task-specific knowledge and human-designed rules.

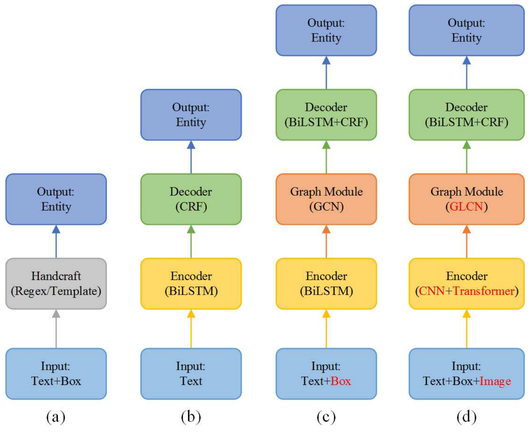

Figure 1: Typical architectures and our method for key information extraction. (a) hand-craft features based method. (b) automatic extraction features based method. (c) using more richer features based method. (d) our proposed models.

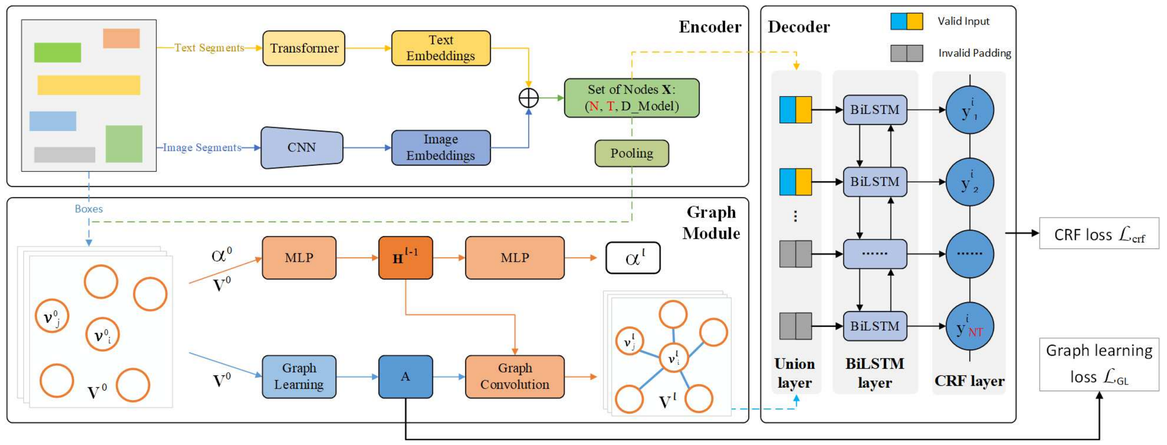

Figure 2: Overview of PICK

Method

The overall architecture is shown in Figure 2, which contains 3 modules:

- *Encoder*: This module encodes textual and morphology information individually, which will be used as node input to the Graph Module.
- *Graph Module*: This module can catch the latent relation between nodes and get richer graph embeddings representation of nodes through improved *graph learning-convolutional operation*, which get non-local and non-sequential features.
- *Decoder*: This module performs sequence tagging on the union non-local sentence at character-level using BiLSTM and CRF, respectively.

Graph Learning

Given an input $\mathbf{V} = [\boldsymbol{v}_1, \dots, \boldsymbol{v}_N]^T \in \mathbb{R}^{N \times d_{model}}$ of graph nodes, where $\boldsymbol{v}_i \in \mathbb{R}^{d_{model}}$ is the i-th node of the graph, Graph Module generate a *soft adjacent matrix* $\mathbf{A}$ that represents the pairwise relationship weight between two nodes.

$$\begin{cases} \mathbf{A}_i = \text{softmax}(\mathbf{e}_i)\,, \quad i = 1, \dots, N, \quad j = 1, \dots, N\,, \\ \text{e}_{ij} = \text{LeakRelu}(\mathbf{w}_i^T |\boldsymbol{v}_i - \boldsymbol{v}_j|)\,, \end{cases} \tag{1}$$

where $\mathbf{w}_i \in \mathbb{R}^{d_{model}}$ is learnable weight vector.

$$\sum_{j=1}^{N} A_{ij} = 1, A_{ij} \geq 0\,. \tag{2}$$

We use the modified loss function to optimize the learnable weight vector $\mathbf{w}_i$ as follows

$$\mathcal{L}_{\text{GL}} = \frac{1}{N^2} \sum_{i,j=1}^{N} \exp(A_{ij} + \eta \|\boldsymbol{v}_i - \boldsymbol{v}_j\|_2^2) + \gamma \|\mathbf{A}\|_F^2\,, \tag{3}$$

where $\|\cdot\|_F$ represents Frobenius-Norm. γ is a tradeoff parameter and larger γ brings about more sparsity *soft adjacent matrix* $\mathbf{A}$ of graph.

Graph Convolution

Firstly, given an input $\mathbf{V}^0 = \mathbf{X}_0 \in \mathbb{R}^{N \times d_{model}}$ as the initial layer input of the graph, initial *relation embedding* $\boldsymbol{\alpha}_{ij}^0$ between the node v_i and v_j is formulated as follows

$$\boldsymbol{\alpha}_{ij}^0 = \mathbf{W}_\alpha^0 [x_{ij}, y_{ij}, \frac{w_i}{h_i}, \frac{h_j}{h_i}, \frac{w_j}{h_i}, \frac{T_j}{T_i}]^T\,, \tag{4}$$

where $\mathbf{W}_\alpha^0 \in \mathbb{R}^{dmodel \times 6}$ is learnable weight matrix.

Then we extract *hidden features* $\mathbf{h}_{ij}^l$ between the node v_i and v_j from the graph using the *node-edge-node* triplets (v_i, α_{ij}, v_j) data in the l-th convolution layer, which is computed by

$$\boldsymbol{h}_{ij}^l = \sigma(\mathbf{W}_{v_i h}^l \mathbf{v}_i^l + \mathbf{W}_{v_j h}^l \mathbf{v}_j^l + \boldsymbol{\alpha}_{ij}^l + \mathbf{b}^l)\,, \tag{5}$$

Finally, *node embedding* $\mathbf{v}_i^{l+1}$ aggregate information from *hidden features* $\boldsymbol{h}_{ij}^l$ using graph convolution to update node representation. For node v_i, we have

$$\mathbf{v}_i^{(l+1)} = \sigma(\mathbf{A}_i \mathbf{h}_i^l \mathbf{W}^l)\,, \tag{6}$$

where $\mathbf{W}^l \in \mathbb{R}^{d_{model} \times d_{model}}$ is layer-specific learnable weight matrix in the l-th convolution layer.

The *relation embedding* $\boldsymbol{\alpha}_{ij}^{l+1}$ in the $l+1$-th convolution layer for node v_i is formulated as

$$\boldsymbol{\alpha}_{ij}^{l+1} = \sigma(\mathbf{W}_\alpha^l \mathbf{h}_{ij}^l)\,, \tag{7}$$

where $\mathbf{W}_\alpha^l \in \mathbb{R}^{d_{model} \times d_{model}}$ is layer-specific trainable weight matrix in the l-th convolution layer.

Results

Table 1: Performance comparison between PICK (Ours) and baseline method on Medical invoice datasets.

Entities	Baseline			PICK (Our)		
	mEP	mER	mEF	mEP	mER	mEF
MIT	66.8	77.1	71.6	85.0	81.1	**83.0**
CCTA	85.7	88.9	87.3	93.1	98.4	**95.6**
IN	61.1	57.7	59.3	93.9	90.9	**92.4**
SSN	53.4	64.6	58.5	71.3	64.6	**67.8**
Name	73.1	73.1	73.1	74.7	85.6	**79.8**
HN	69.3	74.4	71.8	78.1	89.9	**83.6**
Overall (micro)	71.1	73.4	72.3	85.0	89.2	**87.0**

Table 2: Results on SROIE and train ticket datasets.

Method	Train Ticket (mEF)	SROIE (mEF)
Baseline	85.4	-
LayoutLM	-	95.2
PICK (Ours)	98.6	96.1

Table 3: Results of each component of our model.

Model	Medical Invoice (mEF)	Train Ticket (mEF)
PICK (Full model)	87.0	98.6
w/o image segments	↓0.9	↓0.4
w/o graph learning	↓1.6	↓0.7

https://github.com/wenwenyu/PICK-pytorch

Contact information: Wenwen Yu, No. 209 Tongshan Road, School of Medical Imaging, Xuzhou Medical University, Xuzhou 221000, China – Phone: (+86)183–6127–7609 – Email: *yuwenwen62@gmail.com*; Web: *www.yuwenwen.site*