Joint Learning with BERT-GCN and Multi-Attention for Event Text Classification and Event Assignment

XIANGRONG SHE, JIANPENG CHEN, and GANG CHEN
Yangtze River Delta Information Intelligence Innovation Research Institute, Wuhu, 241000, China

Corresponding author: GANG CHEN (e-mail: cheng@ustc.win).

This work was supported in part by the Key Research and Development Plan of Anhui Province under Grant 202104a05020071.

ABSTRACT Government hotline is closely related to people’s lives and plays an important role in solving social problems and maintaining social stability in China. However, the event text of the hotline is inconsistent in length and unclear in elements, so it is a challenge for the operator to manually complete the assignment tasks of hotline event. To address these problems, we propose a joint learning method for event text classification and event assignment for Chinese government hotline. Firstly, graph convolution network (GCN) and BERT are used to process the event text respectively to obtain the corresponding representation vector. Then, the obtained two representation vectors are fused by the dynamic fusion gate to get fusion vector and classified the fusion vector through the text classification. Secondly, we use multi-attention mechanism to process the GCN result vector, BERT result vector and the “sanding” vector to obtain attentive “event-sanding” representation vector and calculate the corresponding department probability distribution. Finally, the historical prior knowledge based reorder model is used to sort the results of the “event-sanding” matching module and output the optimal assignment department of government hotline event. Experimental results show that our method can achieve better performance compared with several baseline approaches. The ablation experiments also demonstrate the validity of each proposed module in our model.

INDEX TERMS Event assignment, event text classification, graph convolution networks, BERT, multi-attention, government hotline

I. INTRODUCTION

Government hotline is an important channel for the government to connect with the public. It is an effective way for citizens to participate in social governance in China [1]. Through the government hotline, the government can learn about people’s livelihood issues in a timely manner, effectively resolve social conflicts, and improve governance capacity and efficiency. In recent years, several local governments in China have made some achievements in the digitalization reform of hotlines, but there are also problems in the event assignment process. As the hotline operator cannot deeply understand the rights and responsibilities of each department, it is still difficult to select the right responsible department from dozens of departments to complete event assignment after fully understanding the needs of citizens, resulting in low accuracy and efficiency of event assignment. Therefore, it is of great significance to develop an automatic assignment method which can accurately determine the corresponding responsible departments.

Different from traditional natural language texts, citizens’ telephone texts have problems of different lengths, and most of them are narrative descriptions, often with unclear descriptions and incomplete elements [2]. Therefore, it is difficult to accurately allocate events by simply using hotline text data. Combining it with public data of government departments (such as department profile and “sanding” data, etc.) can provide support for effectively allocating various events. It is worth noting that the “sanding” includes the organization’s specifications, main responsibilities, internal organizations and their specific responsibilities, staffing and leading positions. Therefore, how to effectively use the “sanding” information to improve the accuracy of event assignment has become an important research focus of event assignment task.
How to effectively process the event text and get the semantic encoding of the event is the core task of event assignment. Traditional word embedding methods such as Word2vec [3] and GloVe [4] can only consider the simple context information of the text, but it is difficult to deal with polysemy in the natural language text. The pre-trained language model (BERT[5], XLNet[6], RoBERTa[7], etc.) can effectively solve this problem. But the pre-trained language model has the limitation of the input text length, and difficult to deal effectively with the long text. Therefore, when process the long event text, it will truncate the input text which will lead to the information loss problem. This problem can be solved by constructing the event graph and applying the GCN [8] to nodes and edges of constructed graph. At the same time, attention mechanism [9] can improve the event assignment accuracy by enhancing the association between hotline events and “sanding” data of government departments. In addition, the event text type can provide a priori knowledge for event assignment to further improve the accuracy of event assignment. Based on above, we proposed a joint learning method for event text classification and event assignment. Our contributions are as follows:

1) We jointly learn event text classification and event assignment to reduce model parameters and improve model accuracy. A novel joint learning framework of event text classification and event assignment is proposed to employ the event question type classification task to guide the event text encoding, and then leverage the event text encoding to conduct event assignment.

2) To solve the problem of different event text length and unclear elements, we propose a BERT-GCN model, which uses the gate mechanism based on prior knowledge to realize the dynamic feature fusion of BERT and GCN, so as to effectively extract event text features to complete the task of event text classification and event assignment.

3) We proposed an “event-sanding” matching model based on multi-attention, it can effectively match the “event-sanding” pairs by applying several attentions to “GCN-sanding”, GCN-BERT and BERT-“sanding”.

II. RELATED WORK

A. GRAPH CONVOLUTIONAL NETWORKS

In recent years, due to the universal existence of graph data, the topic of graph neural network (GNN) has attracted more and more attention of researchers [10, 11]. A number of researchers apply existing neural network models to graph structure to achieve end-to-end graph modeling [8, 12, 13], and graph convolution neural network is the most active one. In the pioneering work, Bruna et al. [14] proposed the first graph convolution neural network in 2013, they defined graph convolution in spectral space based on graph theory and convolution theorem. The original spectral method has the disadvantage of high space-time complexity, Kipf and Welling [8] presented a simplified graph neural network model, called graph convolutional networks (GCN), which can reduce the complexity of space-time greatly through parameterize the convolution kernel in spectral method. Similarly, Defferrard et al [15], proposed a CNN algorithm based on spectral graph theory, which can generalize convolutional neural networks from low-dimensional regular grids to high-dimensional irregular domains. It also can reduce the complexity of space-time greatly. Besides to the above work, GCN was also applied in several NLP tasks such as semantic role labeling [16], relation classification [17] and machine translation [18], where GCN is used to encode syntactic structure of sentences. For text classification tasks, there is also a lot of work on graph neural networks [8, 12, 13, 19, 20]. However, they only use word as nodes to construct graph in a document or a sentence, which will lose the semantic information of the text. Therefore, we focus on combining language models with GCN by a dynamic fusion gate, which can effectively utilize the textual prior knowledge.

B. TEXT CLASSIFICATION

Text classification refers to extracting features from original text data and predicting the categories of text data based on these features. Text classification can be applied to many natural language processing (NLP) tasks, such as emotion analysis, topic tagging, question answering, and conversational behavior classification. For machine learning models, NB was the first model to be used for text classification task. Subsequently, such as KNN, SVM, RF, XGBoost[21], LightGBM[22] and other machine learning algorithms have been widely used for text classification.

For deep learning model, Kim proposed TextCNN[23], which firstly use CNN for sentence classification. Tang et al. [24] used CNN or LSTM to extract single sentence representation, and then used gated-RNN to encode the internal relationships and semantic relationships between sentences, so as to obtain document-level text coding for text classification. However, CNN model can only extract local features of text effectively, and it is difficult to obtain contextual semantic information of text. Although LSTM and GRU models can extract contextual semantic information, they cannot extract local features of sentences well. In order to give full play to the respective advantages of CNN and RNN, more and more researchers combine CNN and RNN [25-27]. Meanwhile, to further increase the representation accuracy of such models, attention mechanisms have been introduced as an integral part of text classification models [28, 29]. Although the combination of CNN and RNN has achieved great results, its essential defects are difficult to change. With the emergence and development of pre-trained language models, such problems have been effectively improved. Pre-trained language models can effectively learn global semantic representation and significantly boost NLP tasks, including text classification. Pre-trained language models have some limitation on the text length, so it is difficult to extract the
syntactic structure features of long text effectively. The good news is the GNN-based models can extract the syntactic structure features of long text by encoding text graph nodes, so some researchers study using GNN for text classification. Hao et al. [19] propose a graph-CNN based model which can convert text to graph-of-words, then use graph convolution operations to convolve the word graph. It has the advantage of learning different levels of semantics with CNN models. Yao et al. [30] propose the text graph convolution network (TextGCN) for text classification, which builds a single text graph for a whole dataset based on word co-occurrence information and document word relations.

C. ANSWER SELECTION

Answer selection as a sub-task of automatic question answering has attracted wide attention of researchers, and neural network-based models have been proved to be effective in this task. In 2015, Severyn et al. [31] use CNN to process query and document to obtain the feature maps respectively, and then a similarity matrix is added to enhance the predict result. Also in 2015, Wang et al. [32] propose an RNN based approach to answer selection problems, which uses a bidirectional Long-Short Term Memory (BiLSTM) network to sequentially read words from question and answer sentences, and outputs the relevance scores. Instead of learning the vector representation of questions and answers separately, recent studies have introduced attention mechanism to highlight the relevant interaction information between questions and answers, which can better focus on relevant parts between them. Wen et al. [33] proposed a hybrid attention mechanism for automatic answer selection, which can align the most informative part of the question-and-answer pair. Yang et al. [34] proposed a hierarchical attention network, which fully fuse the input documents and knowledge base by taking advantage of the semantic characteristics of input sequences. With the widespread application of pre-trained language model, the accuracy of answer selection task can also be improved by pre-trained method. Laskar et al. [35] used pre-trained language model BERT for answer selection tasks, which can effectively leverage the context of each word in a sentence and improve performance of the model.

III. METHOD

A. PROBLEM DEFINITION

We aim to jointly conduct two tasks: event text classification and event assignment, which can be regard as a domain specific text classification problem and text matching problem. Given an event description text $T_i$, the goal is to simultaneously classify the correct type from a candidate set $C = \{c_1, c_2, ..., c_n\}$ and select the correct department for a candidate set $D$ through a specific answer selection task. The specific answer selection task is to select the set of correct answers from a set of candidates $S_i = \{s_i^1, s_i^2, ..., s_i^n\}$, and then ranking the candidates to get the best one.

![Diagram](https://example.com/figure1.png)

**FIGURE 1.** Event assignment model structure

B. MODEL

We introduce the proposed joint learning model for event text classification and event assignment. As is depicted in figure 1, The overall framework consists of three components: (1)GCN...
and BERT based Event Text Classification module, (2) Multi-
attention based “event-sanding” matching module, (3) reordering module. Firstly, module (1) is used to encode the input event text to obtain $V_{GCN}$ and $V_{Lin}$ vectors, and then two vectors are fused through a specially designed gate to conduct a text classification. Secondly, the attentive “event-sanding” representation vector is calculated by $V_{GCN}$ and $V_{Lin}$ vectors together with the “sanding” vector $V_{Sanding}$ in module (2) through a multi-attention layer. The representation vector is then used to calculate the “event-sanding” matching probability. Finally, module (3) is used to reorder all the calculated matching probabilities to get the final assigned department.

1) GCN AND BERTBASED EVENT TEXT CLASSIFICATION

The event classification model we proposed includes 4 modules: (i) Graph Construction; (ii) Graph Convolution Network; (iii) BERT layer; (iv) Dynamical Fusion layer.

Graph Construction. In this section, we introduce how to construct the event key information interaction graph from an event text. As the event text has a length problem and some sentences in the event text are not relevant to the event topic, Therefore, we extract the keywords of the event text which serve as the topics of the event. In the next steps, these keywords are service as graph node, which can represent the key information of event graph. Since keyword extraction is not the main point of this paper, we use the exist tools to extract it. We adopted two methods to construct event graph between sentences and within sentences respectively. Algorithm 1 shows the construction process of event graph between sentences.

**Algorithm 1 construction of event graph between sentences**

**Input:** The event text $D$  
**Output:** Event graph between sentences  
1. Do word segmentation, named entity recognition and keyword extraction on $D$ to get the keywords $K$  
2. for sentence $s$ in $D$ do  
3. if $s$ contains a keyword then  
4. Assign $s$ to vertex $v_s$  
5. else  
6. Assign $s$ to vertex $v_{empty}$  
7. end if  
8. end for  
9. for vertex $v_i$ and $v_j$ do  
10. if $v_i$ and $v_j$ in the same sentence then  
11. add an edge between $v_i$ and $v_j$  
12. else  
13. continue  
14. end if  
15. end for

Given an event text, we first use the exist tool to do word segmentation, named entity recognition and keyword extraction on it. After get the keywords of event, we can use these keywords to construct a coarse graph. As shown in Algorithm 1, for the $k$ keywords extracted, we first adopt a simple strategy that assigns a sentence $s$ to the keyword $k$ if $k$ appears in the sentence. Note that one sentence can be associated with multiple keywords, which implicitly indicates there exist connection between them, so we add an edge between them. Sentences that do not contain any of the keywords are put into a special node called “empty”.

**Algorithm 2 construction of event graph within sentence**

**Input:** The event sentence $s$ and the given keyword $k$ in $s$, the distance threshold $L$  
**Output:** Event graph within sentence  
1. Do dependency parsing on sentence $s$ to obtain the syntactic dependency graph  
2. for word in sentence $s$ do  
3. if distance$(w, k)$ <= $L$ graph then  
4. add an edge between $w$ and $k$  
5. else  
6. continue  
7. end if  
8. end for

After this, we have constructed a coarse graph between sentences of events, but lack of relations and nodes within a sentence. Therefore, in the next steps we will show the construction process of event graph within sentences. As shown in Algorithm 2, for the given keyword and corresponding sentence, we first use the dependency parsing method (such as HIT LTP) to analyze the sentence and obtain the corresponding syntactic dependency graph. Then, all words whose distance from the given keyword is less than $L$ in the syntactic dependency graph are retrieved as the nodes inside the sentence, and an edge is added between all retrieved word and the keyword. The edge weight of the retrieved word and the keyword is the distance between them. Together with the coarse graph constructed by Algorithm 1, the event graph of the given event text is completed.

Graph Convolution Network. For the event text $D$ processed by word segmentation, we first implement word embedding operation on each token in $D$ to get the word embedding sequence,

$$Q = \{e_1, e_2, ..., e_i, ..., e_j, ...\}$$

where $e^k_i$ is the keyword embedding, $e^i$ and $e_j$ share the same embedding table. To represent position information of each word, the positional embedding is added to each word in the event text. The corresponding position embedding sequence is as follows,

$$P = \{p_1, p_2, ..., p_i, ..., p_j, ...\}$$

where $p_j$ is the non-keyword embedding and $p_i$ is the keyword embedding. The final embedding of keyword and non-keyword is the sum of their corresponding word embedding and position embedding respectively.

$$e^k_i = e^i + p_i$$

$$e^k_i = e^i + p_i$$

where $e^i$ and $e^k_i$ are the final embedding vectors corresponding to non-keyword and keyword respectively, so the final embedding result of event text $D$ is as follows,

$$E = \{e^k_1, e^k_2, ..., e^k_i, ..., e^k_j, ...\}$$

For the final embedding set $E$, we use self-attention mechanism to get the keyword representation vector $v_j$ that contains context information, the formula is shown in (1)-(4),

$$v_j = \alpha^k j e^k_j$$  

(1)
After we get the vertex representation vector set \( V = \{v_1, v_2, \ldots, v_i, \ldots\} \) in the graph, we feed them to a GCN to make use of the graph structure of the constructed graph. We use an implementation of GCN model similar to the work of Kipf and Welling [8]. So, the GCN follows the following layer-wise propagation rule:

\[
H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)})
\]

where, \( A \) is adjacency matrix of the undirected graph, \( I_N \) is the identity matrix, \( W^{(l)} \) is a layer-specific trainable weight matrix. \( \sigma(\cdot) \) denotes an activation function (such as ReLU). \( H^{(0)} \in \mathbb{R}^{N \times D} \) is the matrix of activations in the \( l \)-th layer; \( H^{(0)} = X \).

For the input vertex representation vector set \( V \), we calculate the vector representation \( V_{\text{gc}} \) of the event text according to formula (8)-(9),

\[
\tilde{A} = \tilde{D}^{-\frac{1}{2}} A \tilde{D}^{-\frac{1}{2}}
\]

\[
V \_{\text{gc}} = \tilde{A} \text{ReLU}(\tilde{A}W^{(0)}W^{(1)})
\]

where, \( W^{(0)}, W^{(1)} \) are input-to-hidden weight matrix and hidden-to-output weight matrix respectively.

**BERT layer.** For the given event text \( D \), BERT is used as the information extraction model to get the event context vector \( V_{\text{lm}} \). Its calculation formula is as follows,

\[
V_{\text{lm}} = BERT(D)
\]

**Dynamical Fusion layer.** To solve the problem of varying length of event text, we use a Dynamical Fusion method based on a gating mechanism to fuse GCN information \( V_{\text{gc}} \), and language model information \( V_{\text{lm}} \) to obtain a fused contextual semantic information representation \( V_{\text{fuse}} \). The calculation formula is as follows:

\[
V_{\text{fuse}} = (\lambda_1 \odot V_{\text{gc}}) + (\lambda_2 \odot V_{\text{lm}})
\]

where, \( \lambda_1, \lambda_2 \) are the fusion weights of GCN information and language model information respectively, it is obtained by combining the prior knowledge of event text and the calculation formula is as follows:

\[
g = \text{sigmoid}(W_g [V_{\text{gc}}, V_{\text{lm}}] + b_g)
\]

\[
\lambda_1 = \begin{cases} 
\frac{l_{\text{test}} - l_{\text{best}}}{l_{\text{test}}} & l_{\text{test}} > l_{\text{best}} \\
0 & \text{else}
\end{cases}
\]

\[
\lambda_2 = \frac{g + \lambda}{2}
\]

where \( W_g \) is the learnable weight matrix, \( b_g \) is the bias vector, and \( \odot \) is the element-wise multiplication. The probability distribution of event text types is calculated as follows,

\[
p_{\text{et}} = \text{softmax}(W_{\text{fuse}} V_{\text{fuse}} + b_{\text{fuse}})
\]

where \( W_{\text{fuse}} \) is the learnable weight matrix, \( b_{\text{fuse}} \) is the bias vector, and \( p_{\text{type}} \) is the event category probability.

2) **MULTI-ATTENTION BASED “EVENT-SANDING” MATCHING**

In order to enhance the interaction between event and “sanding”, we propose a multi-attention mechanism to fetch the important information from GCN-event result, BERT-event result and BERT-“sanding” result.

**GCN-event and BERT-“sanding” attention.** For the given “sanding” text \( D_{\text{sanding}} \), we use BERT model to extract its context information and obtain the corresponding representation vector \( V_{\text{sanding}} \).

\[
V_{\text{sanding}} = BERT(D_{\text{sanding}})
\]

We use the two-way attention mechanism [36] to model the interaction between GCN-event and BERT-“sanding”, and the specific formula is as follows,

\[
M_{\text{ges}} = \tanh(V_{\text{ges}}^T U_{\text{ges}} V_{\text{sanding}})
\]

where \( U_{\text{ges}} \in \mathbb{R}^{d_{\text{ges}} \times d_{\text{sanding}}} \) is the learnable weight matrix; \( d_{\text{ges}} \) and \( d_{\text{sanding}} \) are the dimensions of GCN-event output vector and BERT-“sanding” output vector respectively.

Then we apply max-pooling over \( M_{\text{ges}} \) row-wise and column-wise to obtain the attention weights for the GCN-event and the BERT-“sanding”:

\[
\alpha_{\text{ges}} = \text{softmax}(\text{Max}(M_{\text{ges}}))
\]

\[
\alpha_{\text{sanding}} = \text{softmax}(\text{Max}(M_{\text{sanding}}))
\]

where \( \alpha_{\text{ges}} \) and \( \alpha_{\text{sanding}} \) are the attention weights for the GCN-event and the BERT-“sanding”. We conduct dot product between the attention vectors and the representation vectors to generate the final attentive representations for GCN-event and BERT-“sanding”:

\[
l_{\text{ges}} = V_{\text{ges}}^T \alpha_{\text{ges}}
\]

\[
\hat{l}_{\text{sanding}} = V_{\text{sanding}}^T \alpha_{\text{sanding}}
\]

**GCN-event and BERT-event attention.** In order to better highlight the internal correlation of events, we concatenate the representation vectors of GCN-event and BERT-event,

\[
V_{\text{concat}} = \text{concat}(V_{\text{ges}}, V_{\text{lm}})
\]

Then the self-attention mechanism is used to process the concatenated representation vector, and the specific formula is as follows:

\[
M_{ee} = \text{LeakyReLU}((V_{\text{concat}})^T U_{ee} V_{\text{concat}})
\]

\[
\alpha_{ee} = \text{softmax}(\text{Max}(M_{ee}))
\]

\[
\alpha_{s} = \text{softmax}(\text{Max}(M_{s}))
\]

where \( U_{ee} \in \mathbb{R}^{d_{\text{concat}} \times d_{ee}} \) is the learnable weight matrix, \( d_{\text{concat}} \) is the dimension of \( V_{\text{concat}} \); \( \text{Max()} \) is max-pooling.
\( \alpha_{s}^{\text{event}} \) and \( \alpha_{s}^{\text{event}} \) are the attention distributions in two dimensions.

We conduct dot product between the attention vectors and the representation vectors to generate the attentive representations and then use a linear layer to get the final representations for GCN-event and BERT-event:

\[
\begin{align*}
    r_{s} &= (V_{\text{event}} \cdot \alpha_{s}^{\text{event}}) \quad (27) \\
    r_{y} &= (V_{\text{event}} \cdot \alpha_{y}^{\text{event}}) \quad (28) \\
    r_{\text{concat}} &= \text{linear}(\text{concat}(r_{s}, r_{y})) \quad (29)
\end{align*}
\]

**BERT-event and BERT-"sanding" attention.** Similar to GCN-event and BERT-"sanding" attention, the two-way attention mechanism is also used to model the interaction between BERT-event and BERT-"sanding", and the specific formula is as follows,

\[
M_{\text{lex}} = \tanh(V_{\text{lex}}^{T} U_{\text{lex}} V_{\text{sanding}}) \quad (30)
\]

where \( U_{\text{lex}} \in \mathbb{R}^{d_{\text{lex}} \times d_{\text{lex}}} \) is the learnable weight matrix, \( d_{\text{lex}} \) and \( d_{\text{sanding}} \) are the dimensions of BERT-event output vector and BERT-"sanding" output vector respectively.

Then we apply max-pooling over \( M_{\text{lex}} \) row-wise and column-wise to obtain the attention weights for the BERT-event and the BERT-"sanding":

\[
\alpha_{\text{lex}} = \text{softmax}(\text{Max}(M_{\text{lex}})) \quad (31)
\]

\[
\alpha_{\text{sanding}}^{2} = \text{softmax}(\text{Max}(M_{\text{lex}}^{T})) \quad (32)
\]

where \( \alpha_{\text{lex}} \) and \( \alpha_{\text{sanding}}^{2} \) are the attention weights for the BERT-event and the BERT-"sanding".

Also similar to the GCN-event and BERT-"sanding" attention, we conduct dot product between the attention vectors and the representation vectors to generate the final attentive representations for BERT-event and BERT-"sanding":

\[
\begin{align*}
    r_{\text{lex}} &= V_{\text{lex}}^{T} \alpha_{\text{lex}} \quad (33) \\
    r_{\text{sanding}}^{2} &= V_{\text{sanding}}^{T} \alpha_{\text{sanding}}^{2} \quad (34)
\end{align*}
\]

In order to make use of the contribution of each attention, we concatenate all of attention representation vectors in this paper. Finally, the attentive "event-sanding" representation vector will be:

\[
V_{\text{att}} = [r_{\text{lex}}^{1}, r_{\text{sanding}}^{2}, r_{\text{concat}}^{1}, r_{\text{lex}}^{2}, r_{\text{sanding}}^{2}] \quad (35)
\]

Then the corresponding probabilities of "event-sanding" matching are acquired through the softmax function:

\[
p_{\text{att}} = \text{softmax}(W V_{\text{att}} + b) \quad (36)
\]

where \( W \) is the weight which needs to be learned; \( b \) is a bias term.

**3) JOINT TRAINING PROCEDURE**

**Event Classification Loss.** In the event classification based on GCN and BERT module, we specify the loss as the cross-entropy between the predicted and ground-truth:

\[
L_{\text{ct}} = -\sum_{j} y_{j}^{s} \log p_{j}^{s} \quad (37)
\]

where \( p_{j}^{s} \) is the predicted classification probability of event \( j \), and \( y_{j}^{s} \) is the indicator variable (0 or 1).

**"event-sanding" Matching Loss.** In the multi-attention based "event-sanding" matching module, the "event-sanding" matching task is trained to minimize the cross-entropy loss function:

\[
L_{\text{att}} = -\sum_{i} [y_{i} \log p_{i} + (1 - y_{i}) \log(1 - p_{i})] \quad (38)
\]

where \( p_{i} \) is the "event-sanding" matching probability, and \( y_{i} \) is the binary classification label of the “event-sanding” pair.

**Overall Loss Function.** In this paper, we use the joint loss function strategy to optimize the whole network, and the final objective function is to minimize the above two loss functions.

\[
L = \lambda_{1} L_{\text{ct}} + \lambda_{2} L_{\text{att}} \quad (39)
\]

where \( \lambda_{1} \) and \( \lambda_{2} \) are the hyper-parameters to balance losses.

**4) PRIOR KNOWLEDGE BASED REORDERING**

After the government hotline events are processed by the model, a list of probability is obtained by matching all "sanding" to one event. Due to the fact that a department contains multiple responsibilities, the matching probability results need to be reordered for one department. In this paper, the reordering module reorders probabilities of the "event-sanding" by using a weighted average strategy, and then to get the department selection. The weighting strategy is acquired based on the prior knowledge of the events.

Since both historical event assignment and new event assignment follow the distribution of event assignment, so there is a high similarity for the probability distribution of the "event-sanding" matching between new and historical events. Therefore, we construct the reordering module, in which the assignment weights of all "sanding" are calculated by the prior knowledge of historical assignment statistics of "sanding". After the "sanding" weights are obtained, the reordering module process the matching values by weighting and averaging, and then calculate the total score for each department. Finally, the best department is selected based on the score.

**IV. EXPERIMENT**

**A. DATASETS**

In this paper we use an experimental dataset based on the real event assignment case of Wuhu government hotline, including two parts of "event-department" and "event-sanding". The dataset of “event-department” is constructed based on the real event assignment processing result of government hotline, containing 30,000 corresponding event data of 30 municipal departments. The dataset of “event-sanding” is manually annotated by service personnel based on the actual processing results of 30,000 pieces of data in the “event-department” datasets, including 30,000 positive samples (matched) and 60,000 negative samples (unmatched). The datasets description is shown in Table 1.

**TABLE I**

**DESCRIPTION FOR DATASET**
B. EXPERIMENT SETTING
In this paper, The PyTorch 1.7.1 is used to build a network model and experiments are carried out on a NVIDIA GeForce GTX 3090 for Ubuntu 18.04 LTS system.

We use BERT-wmnl[5] model as our language model to carry out semantic representation of event and “sanding”, the language model embedding dimension is 768, vocabulary size is 30000 and input sequence length is 512. The GCN embedding size is set to 768. The Adam optimizer with a learning rate of 10^{-5} is used as the optimization method of the model. We train our models in batches with size of 16.

C. EVENT TEXT CLASSIFICATION RESULT
We first compare the proposed method with several state-of-the-art methods on the text classification task, including HAN[28], TextGCN[30], XLNet[6], BertGCN[37]. Micro-F1 and Weighted-F1 are adopted as evaluation metrics.

| models       | Micro-F1 | Weighted-F1 |
|--------------|----------|-------------|
| HAN          | 0.5067   | 0.4353      |
| TextGCN      | 0.5522   | 0.4888      |
| XLNet        | 0.6131   | 0.5646      |
| BertGCN      | 0.6255   | 0.5778      |
| Ours         | 0.6508   | 0.6075      |

Table 2 summarizes the experimental results of different methods on event text classification. We show that the joint learning model achieves state-of-the-art performance on the event classification task. There are several notable observations in the results. (i) HAN only uses BiLSTM network and multi-level attention mechanism to extract text features, which can extract text contextual information only, but is poor in extracting text local features and has limitations on the length of input text; (ii) When GCN is used to encode the text graph, it can effectively obtain the syntactic structure information of the text and solve the text length problem better. However, it has poor ability to extract the text context information. (iii) Language model can effectively encode text context information, but it also has limitations on the length of the input text; (iv) In this paper, we combine language model with GCN through a gating fusion mechanism which is specific designed for the task, it can effectively extract the effective information of event text, so our model got the optimal effect. The experimental results in Table 3 further prove the correctness of the above analysis and the validity of the proposed model.

D. EVENT ASSIGNMENT RESULT
To evaluate the event assignment, we also compare the proposed method with the following state-of-the-art baseline methods on event assignment task, including Siamese-BiLSTM-based[38], ABCNN-based[39], BERT-BiGRU-based[5], ELECTRA-BiGRU-based[40]. In our paper, the first five results accuracy (P@5), mean average precision (MAP) and mean reciprocal ranking (MRR) indexes are used to measure the overall effect of the answer selection network in the “event-sanding” matching module. Precision, Recall and F1 indexes were used to evaluate the performance of event assignment.

| models                      | P@5  | MAP  | MRR  | Precision | Recall | F1   |
|-----------------------------|------|------|------|-----------|--------|------|
| Siamese-BiLSTM-based        | 0.8459 | 0.8570 | 0.8486 | 0.8688   | 0.8363 | 0.8507 |
| ABCNN-based                 | 0.8251 | 0.8262 | 0.8574 | 0.8654   | 0.8621 | 0.8593 |
| BERT-BiGRU-based           | 0.8310 | 0.8300 | 0.8681 | 0.8628   | 0.8643 | 0.8600 |
| ELECTRA-BiGRU-based        | 0.8389 | 0.8521 | 0.8414 | 0.8659   | 0.8355 | 0.8476 |
| Ours                        | 0.8587 | 0.8725 | 0.8767 | 0.9151   | 0.8982 | 0.9046 |

The experiment results for event assignment are shown in Table 3. Our method is better than all other baseline methods in the effect of each indicator, it has a 3% to 5% improvement in overall event assignment performance. In the “event-sanding” matching module, the P@5, MAP, MRR indicators of our method improved by about 1% to 3%. Compared with Siamese-BiLSTM-based model and ABCNN-based model (use BiLSTM network and CNN as the basic model of feature extraction respectively), we use language model as the feature extraction model, which can effectively obtain the semantic information of text context, and achieve the optimal model effect. Compared with the BERT-BiGRU-based model and ELECTRA-BiGRU-based model (combine language model with BiGRU network for event assignment), we combine language model with GCN, which can effectively solve the input event text length problem. In addition, multi-attention mechanism was used to deal with the matching problem of “event-sanding” pairs, which can effectively highlight the correlation between “event-sanding” pairs. Therefore, compared with other baseline models, our model achieves the optimal assignment effect.

E. ABLATION ANALYSIS
To validate the effectiveness of event text classification module and “event-sanding” matching module in the proposed
method, we also conducted ablation experiments on these three modules respectively.

1) EVENT TEXT CLASSIFICATION MODULE

Our model takes advantages of the GCN and language model, which can effectively deal with the different length problem of input event text. Therefore, two ablation experiments are conducted on the GCN and language model to demonstrate the validity of the two model. During the ablation of GCN, the GCN was removed and only language model was used to extract event text features, while other parts remained unchanged. When ablation of the language model, BERT model was removed and only GCN was used to extract the syntactic structure features of the text, while other parts remained unchanged.

| Model       | Micro-F1 | Weighted-F1 | P@5  | MAP     | MRR    | Precision | Recall | F1     |
|-------------|----------|-------------|------|---------|--------|-----------|--------|--------|
| no-GCN      | 0.6262   | 0.5805      | 0.8543 | 0.8741  | 0.8684 | 0.9080    | 0.8849 | 0.8936 |
| no-LM       | 0.5585   | 0.4913      | 0.8486 | 0.8312  | 0.8671 | 0.8968    | 0.8732 | 0.8761 |
| ours        | 0.6508   | 0.6075      | 0.8587 | 0.8725  | 0.8767 | 0.9151    | 0.8982 | 0.9046 |

Table 4 shows the ablation experiment results of the event text classification module, where “no-GCN” is the results of the ablation GCN and “no-LM” is the results of the ablation language model. It can be seen that our method outperforms the two ablation methods in all evaluation indicators, with an average of more than 3 percentage points. For the GCN ablation model, the “no-GCN” network cannot extract the text context information efficiently and accurately due to the insufficient encoding ability of GCN to the text context information, so the experimental results are inferior to those of the model without ablation. As for the model that ablates language model, it is difficult to extract the syntactic structure information of longer text due to the length problem of the input text, so the “no-LM” results are not as good as those of the model that does not ablate language model. The results of ablation experiment demonstrate the effectiveness of the proposed event text classification module for hotline event assignment.

2) "EVENT-SANDING" MATCHING MODULE

The ablation of “event-sanding” matching module is the ablation of 3 attention module:

① remove GCN-event and BERT-“sanding” attention, only concatenate the other two attention result vector, and the rest remains unchanged.

② remove GCN-event and BERT-event attention, only concatenate the other two attention result vector, and the rest remains unchanged.

③ remove BERT-event and BERT-“sanding” attention, only concatenate the other two attention result vector, and the rest remains unchanged.

Table 5 shows the “event-sanding” matching module ablation results, in which ① is the assignment method for GCN-event and BERT-“sanding” attention module ablation, ② is the assignment method for GCN-event and BERT-event attention module ablation and ③ is the assignment method for remove BERT-event and BERT-“sanding” attention module ablation. As can be seen from the values in Table 6, the model effect is ranked as follows: ours>②>①>③. This is because language model can effectively process most of the input event text, which also conforms to the input length limitation of language model (512) for most of the event text, so the ablation experiment ③ has the worst effect. However, the GCN in this paper is not effective in short text processing, and it is difficult to extract the context information effectively. Moreover, the ablation experiment ② only remove attention layer of a same event with different encode, so the ablation experiment ② is second only to the proposed model. However, ablation experiment ① removes the attention layer between GCN and “sanding” encode, which makes it difficult to effectively use the syntactic structure information of long event text. Therefore, ablation experiment ① is less effective than ablation experiment ②. Based on the above-mentioned experiments, the “event-sanding” matching module we proposed in this paper can effectively improve the accuracy of event assignment.

V. CONCLUSION

In this paper we propose a joint learning method with BERT-GCN and multi-attention for event text classification and event assignment for Chinese government hotline. Firstly, the graph constructing algorithm is used to process the event text to obtain the event graph. Then, the syntactic structure information of event text is obtained by the GCN. At the same time, the context information of event text is obtained by the
BERT model. The two information is fused by the dynamic fusion gate to obtain fusion vector and finally classified the fusion vector through the text classification. Secondly, we use multi-attention mechanism to process the GCN result vector, BERT result vector and the “sandig” vector to obtain five attentive “event-sanding” representation vector. After that, five “event-sanding” representation vector is concatenated and calculate the corresponding department probability distribution. Finally, we use the historical prior knowledge of event distribution to construct a reorder model to sort the results of the “event-sanding” matching module, and finally output the optimal assigned department of government hotline event. The experimental results show that our model achieves SOTA results in many indicators. The ablation experiment also demonstrates the effectiveness of each proposed module.

REFERENCES
[1] L. Ma, Y. P.Zheng, and C. W.Zhang,“The big data empowering effect of government hotlines on city governance innovation: values, status and issues,” Documentation, Information & Knowledge, vol. 2, pp.4-12, 2021.
[2] X.Peng, Y.Liang, and L. Y.Xu,“An Approach for Discovering Urban Public Management Problem and Optimizing Urban Governance Based on 12345 Citizen Service Hotline,” Acta Scientarium Naturalium Universitatis Pekinensis., vol. 56, no. 4, pp.721-731, 2020.
[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” in Proc. ICLR, 2013.
[4] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in Proc. EMNLP, pp. 1532–1543, 2014.
[5] J.Devlin et al., “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” arXiv preprint arXiv:1810.04805, 2018.
[6] Z.Yanget al., “XLNet: generalized autoregressive pretraining for language understanding,” In Proceedings of the 31st Conference on Advances in Neural Information Processing Systems, pp.5754-5764, 2019.
[7] Y. H.Liuet al., “RobERTa: a robustly optimized BERT pretraining approach,” arXiv preprint arXiv:1907.11692, pp.1-13, 2019.
[8] T. N. Kipf, M. Welling, “Semi-supervised classification with graph convolutional networks,” arXiv preprint arXiv:1609.02907, 2016.
[9] A. Vaswani et al., “Attention Is All You Need,” In Proceedings of the 31st International Conference on Neural Information Processing Systems, pp.6000-6010, 2017.
[10] H.Cai et al., “A comprehensive survey of graph embedding: problems, techniques and applications,” IEEE Transactions on Knowledge and Data Engineering, vol. 30, no. 9, pp. 1616–1637, 2017.
[11] P.Battaglia et al., “Relational inductive biases, deep learning, and graph networks,” arXiv preprint arXiv:1806.01261, 2018.
[12] M.Henaff, J.Bruna, and Y.LeCun, “Deep convolutional networks on graphstructured data,” arXiv preprint arXiv:1506.05163, 2015.
[13] M.Defferrard, X.Bresson, and P.Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” In NIPS, pp. 3844–3852, 2016.
[14] J.Bruna, W.Zaremba, A.Szlam, and Y.LeCun, “Spectral networks and locally connected networks on graphs,” In ICLR, 2014.
[15] A.Grover, and J.Leskovec, “node2vec: Scalable feature learning for networks,” Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, USA, pp. 855-864, 2016.
[16] D.Marcheggiani, and I.Titov, “Encoding sentences with graph convolutional networks for semantic role labeling,” In EMNLP, pp. 1506–1515, 2017.
[17] Y.Li, R.Jin, and Y.Luo, “Classifying relations in clinical narratives using segment graph convolutional and recurrent neural networks (seggcnrs),” Journal of the American Medical Informatics Association, 2018. DOI: 10.1093/amiajoc/157.
[18] J.Bastings, I.Titov, W.Aziz, D.Marcheggiani, and K.Simaan, “Graph convolutional encoders for syntax-aware neural machine translation,” In EMNLP, pp. 1957–1967, 2017.
[19] H.Peng, J.Li, Y.He, Y.Liu, M.Bao, L.Wang, Y.Song, and Q.Yang, “Large-scale hierarchical text classification with recursively regularized deep graph-cnn,” In WWW, pp. 1063–1072, 2018.
[20] Y.Zhang, Q.Liu, and L.Song,”Sentence-state lstm for text representation,” In ACL, pp. 317–327, 2018.
[21] X. Chen and C. Guerin, “Xgboost: A scalable tree boosting system,” in Proc. ACM SIGKDD, pp. 785–794, 2016.
[22] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Liu, “Lightgbm: A highly efficient gradient boosting decision tree,” in Proc. NeurIPS, 2017, pp. 3146–3154, 2017.
[23] Y. Kim, “Convolutional neural networks for sentence classification,” in Proc. EMNLP, 2014, pp. 1746–1751, 2014.
[24] D.Tang, B.Qin, and T.Liu, “Document modeling with gated recurrent neural network for sentiment classification,” Proc of the 2015 Conf on Empirical Methods in Natural Language Processing. Stroudsburg, PA: ACL, pp. 1432–1435, 2012.
[25] X.Wang, W.Jiang, and Z.Luo, “Combination of convolutional and recurrent neural network for sentiment analysis of short texts,” Proc of the 26th Int Conf on Computational Linguistics.New York: ACM, pp. 2428–2437, 2016.
[26] Y. Zhang et al., “YNU-HPC: An Emotions at Emoti-2017: Using a CNN-LSTM model for sentiment intensity prediction” Proc of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. Stroudsburg, PA: ACL, pp. 200-204, 2017.
[27] A. M. Alaybaulet al., “A combined CNN and LSTM model for Arabic sentiment analysis” Proc of the Int Cross Domain Conf for Machine Learning and Text Extraction. Berlin: Springer, pp. 179-191, 2018.
[28] Z.Yang, D.Yang, C.Dyer, X.He, A.Smola, and E.Hovy,”Hierarchical attention networks for document classification,” In NAAACL, pp. 1408–1409, 2016.
[29] Y.Wanget al., “Attention-based lstm for aspect-level sentiment classification,” In EMNLP, pp. 606–615, 2016.
[30] L. Yao, C. Mao, and Y. Luo, “Graph convolutional networks for text classification,” in Proc. AAAI, 2019, pp. 7370–7377, 2019.
[31] A. Severyn, and A.Moschitti, “Learning to rank short text pairs with convolutional deep neural networks,” In SIGIR, pp. 373–382, 2015.
[32] D. Wang, and E.Nyberg, “A long short-term memory model for answer sentence selection in question answering,” In ACL, pp. 707–712, 2015.
[33] J. H.Wenet et al., “Hybrid attentive answer selection in CQA with deep users modelling,” In Proceedings of the 32th AAAI Conference on Artificial Intelligence, pp.2556-2563, 2018.
[34] M. Yang et al., “Knowledge-enhanced hierarchical attention for community question answering with multi-task and adaptive learning,” In Proceedings of the 2019 International Joint Conference on Artificial Intelligence, pp. 5349-5355, 2019.
[35] M.Laskar, E.Hoque, and J. X.Huang,”Utilizing bidirectional encoder representations from transformers for answer selection,” International Conference on Applied Mathematics, Modeling and Computational Science, pp. 693-703, 2019.
[36] D. Santos, C. N. Tan, M. Xiang, B. and B.Zhou,”Attentive pooling networks,” CoRR, abs/1602.03609, 2016.
[37] Y.Lin et al., “BertGCN: Transductive Text Classification by Combining GNN and BERT,” arXiv preprint arXiv:2105.05727, 2021.
[38] J.Mueller, and A. Thiyagarajan, “Siamese recurrent architectures for learning sentence similarity,” in AAAI, 2016.
[39] W.Yinet al., “ABCNN:Attention-Based Convolutional Neural Network for Modeling Sentence Pairs,” Transactions of the Association for Computational Linguistics, vol. 4, pp.259–272, 2016.
[40] D.Cortiz,“Exploring transformers in emotion recognition: a comparison of BERT, DistilBERT, RoBERTa, XLNet and ELECTRA,” arXiv preprint arXiv:2104.02041, 2021.
XIANGRONG SHE was born in 1989. He received the master’s degree from Hefei University of Technology, China in 2019. He is working as assistant engineer in Big Data Labs of Yangtze River Delta Information Intelligence Innovation Research Institute from 2020. His research interests include natural language processing and data mining.

JIANGPENG CHEN was born in 1995. He received the master’s degree from University of Science and Technology of China, China in 2020. He is working as assistant engineer in Big Data Labs of Yangtze River Delta Information Intelligence Innovation Research Institute from 2020. His research interests include artificial intelligence, knowledge graph and its application.

GANG CHEN was born in 1982. He received the Ph.D. degree in computer science with Fudan University, China in 2012. He is working as associate researcher in Big Data Labs of Yangtze River Delta Information Intelligence Innovation Research Institute from 2019. His research focus includes data mining and natural language processing.