Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 5, No. 2, 2023

Joint Modelling of Slot Filling and Intent Detection in Constrained Resource Scenarios

Qixuan Li *

School of Computer Science and Technology, Jilin University, Changchun, Jilin, 130015, China
*** Corresponding author Email:** liqx2120@mails.jlu.edu.cn

Abstract: In the era of pervasive smart devices, natural language understanding (NLU) holds a pivotal role for facilitating intelligent interactions and decision-making. Core to NLU are slot filling and intent recognition, essential tasks for comprehending user input. While joint modelling of these tasks has gained prominence, the challenges of realizing efficient joint models on resource-constrained devices have emerged as significant. These devices possess limited computational capacity and real-time requirements, necessitating lightweight and efficient models. In this study, we explore the design of a resource-efficient joint model for slot filling and intent recognition. Through leveraging BERT, $BiLSTM$, graph neural networks, and mask mechanisms, our model achieves the dual goals of semantic slot prediction and intent classification. We focus on model design, training, and real-time inference, aiming to contribute to the paradigm of resource-constrained natural language understanding. Our investigation demonstrates the efficacy of our approach, even when working with a reduced dataset, underscoring the model's applicability to real-world scenarios with limited resources.

Keywords: Slot Filling; Intent Detection; Graph Neural Networks.

1. Introduction

In today's digital era, the widespread use of smart devices has led to the rapid development of the field of natural language understanding, in which slot filling and intent recognition play a key role as core tasks to enable intelligent interaction and decision making. Joint multitasking of slot filling and intent detection is a popular research area in the field of natural language understanding, which aims at identifying semantic slots and their corresponding intents as a summary of the whole sentence. However, in the face of practical scenarios such as resource-constrained mobile devices and embedded systems, how to efficiently conduct joint modelling research on slot filling and intent recognition on such devices has become an important and challenging topic.

However, resource-constrained device environments pose many challenges. These devices usually have limited computing power and storage resources to host complex deep learning models. At the same time, real-time requirements place higher demands on the efficient performance of the models. How to design lightweight and efficient joint models for slot filling and intent recognition tasks under such dual constraints has become a problem that needs to be explored in depth.

For the study of joint models for NLU, the earliest work used a three-layer CRF, where the three layers are token features, slot labels, and intent labels, whereas this architecture is better than performing two sub-tasks in a pipeline, and the first neural model for solving the joint task is $RecNNs$ (different from recurrent neural networks) .[1,2] In recent years, Recurrent Neural Network (RNN) based approaches, especially Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTMs), have demonstrated strong capabilities in dealing with sequence annotation and sentence classification problems. [3-5] However, the prediction accuracy of deep neural network models relies heavily on a large amount of supervised data, and good device and GPU resources also affect the training and performance of the models to a certain extent, therefore, this study aims to explore how to build a joint model of slot filling and intent recognition on resource-constrained devices to achieve efficient natural language understanding in real-world applications. We will focus on the key aspects of model design, training and real-time inference, aiming to provide new solutions for natural language understanding tasks in resource-constrained environments. In this paper, we first reduce the dataset by random sampling to shorten the training period and simulate the resource-constrained scenario, and propose a $BiLSTM$-based model using BERT, graph neural network and masking mechanism in order to accomplish the joint task of slot labelling and intent labelling.

2. Related Works

In this section, we present some related work, mainly on the two models that have been proposed in association with our model.

2.1. Graph Neural Networks for Joint Slot Filling and Intent Detection

Recent advancements in Natural Language Processing (NLP) have led to the development of innovative models to address the challenges of joint slot filling and intent detection tasks. Traditional approaches like linear chain Conditional Random Fields (CRFs) have demonstrated limitations in capturing context and handling complex dependencies, particularly for joint tasks. In response, Tang et al. introduced a Graph Convolutional Network (GCN)-based CRF model to overcome these limitations. [6] The proposed GCN-CRF leverages graph convolutions to model intricate relationships among words, slot labels, and intent labels. By doing so, they successfully enhanced the performance of both slot filling and intent detection, providing a comprehensive solution to the joint task.

2.2. Enhancing Sequence Labeling with Graph Neural Networks

Parallelly, the challenges posed by long-tail scenarios in sequence labelling (SL) tasks have been tackled by Wang et al. [7] They proposed the Graph Neural Network for Sequence Labelling (GNN-SL), a technique that employs Graph Neural Networks (GNNs) to enhance SL model outputs by retrieving similar tag instances from the training dataset. The use of GNNs in constructing a heterogeneous graph allows for the propagation of information between retrieved tag samples and input word sequences. Through enhanced node aggregation and neighbourhood information, their approach effectively addresses the challenges of long-tail SL tasks, improving the overall quality of predictions.

2.3. Fusion of Graph Neural Networks and GCN-CRF Model

Motivated by the strengths of both the GCN-CRF model for joint tasks and the GNN-SL approach for SL tasks, we propose an innovative fusion that unifies the advantages of both paradigms. In this approach, we incorporate the GNN-SL's graph-based enhancement strategy into the GCN-CRF model for joint slot filling and intent detection. By integrating GNN techniques into the existing GCN-CRF framework, we aim to harness the power of relationship modelling and sequence labelling enhancement simultaneously. This fusion is anticipated to result in a more robust and comprehensive model that excels in solving the combined challenges of slot filling and intent detection in NLP applications.

3. Methodology

In this section we describe our dataset selection and preprocessing and our model construction.

3.1. Dataset Description and Preprocessing

The Snips dataset stands as a pivotal resource in the realm of natural language understanding (NLU) research. Specifically curated for intent recognition and slot filling tasks, this dataset encompasses a wide array of languages and domains, facilitating the training and evaluation of diverse NLU models.

3.1.1. Dataset Overview

Comprising user-generated text inputs, the Snips dataset is meticulously annotated with corresponding intent labels and slot annotations. The intents encapsulate the users' purposes behind their inputs, while the slots entail critical information extracted from the utterances, encompassing entities such as dates, times, and locations.

3.1.2. Resource Constraints and Preprocessing

In order to adapt the Snips dataset to resource-limited devices, we introduce a preprocessing step - random sampling. This approach aims to simplify computational requirements while maintaining data diversity. By strategically selecting subsets of the dataset, preprocessing effectively optimizes the dataset for constrained environments. During the preprocessing stage, a random sampling strategy was used to select a representative subset from the Snips dataset, of which we sampled 15%. This process entailed randomizing the selection of data instances while ensuring a proportional distribution of intent and slot annotations. By reducing the size of the dataset through random sampling, we address the challenges posed by limited computational resources on resource-constrained devices.

3.2. Model Construction

We propose an integrated model that aims to provide an efficient and high-performing solution for natural language understanding tasks such as intent detection and slot filling. Our model combines BERT word embeddings, bi-directional long and short-term memory networks ($BiLSTM$), self-attention mechanisms, and graph neural network techniques to form a multi-layered architecture that achieves the goals of deeper comprehension of input text, efficient feature extraction, and accurate result outputs.

3.2.1. BERT

$Bidirectional\ Encoder\ Representation$ (BERT) based on the Transformer architecture has demonstrated powerful capabilities in several tasks in various domains. In this study, we used BERT word embeddings as the word vector representation of the neural network input.[8] For each word in the sentence, we employ $PieceTokenizer$ to further slice it and integrate the sliced representation through a simple summation operation, thus bringing the sliced word back to the original integrated word form. For our experiments, we used the basic version of BERT.

3.2.2. BiLSTM

Bidirectional Long Short-Term Memory (BiLSTM) is a variant of Recurrent Neural Network (RNN) for processing sequential data, especially in the fields of natural language processing and temporal data analysis. It is capable of capturing long-range dependencies when processing sequential information, leading to a better understanding of contextual information.

LSTM is a special type of RNN designed to solve the problem of gradient vanishing during long sequence training. It effectively captures and transmits information through memory cells and a series of gate mechanisms. However, traditional LSTMs can only transfer information in one direction (front-to-back or back-to-front). $BiLSTM$ extends this by processing input sequences in two directions (front-to-back and back-to-front) simultaneously, thus allowing the model to consider both past and future contexts at each time step.

We denote the $i-th$ word as w_i .When all tokens are handled by BERT, the BERT embedding of the $i-th$ word as hd_i . Bidirectional LSTMs are used for word embedding after BERT:

$$\widehat{hd}, \overleftarrow{hd} = BiLSTM\left(\left[\overrightarrow{hd}, \overleftarrow{hd}\right]\right) \quad (1)$$

where $hd = \{hd_0, \dots\}$ is the inputs, and $\rightarrow$ on the top denote the forward order of original embeddings, $\leftarrow$represent the backward order, respectively. $\widehat{hd} = \left\{\widehat{hd}_0, \dots\right\}$ is the outputs of $BiLSTM$. $\overleftarrow{hd}$ is the last state of $BiLSTM$ encoder.

3.2.3. Self Attention

The model uses deep self-attention mechanism in predicting the slot labels to identify the most informative words, while the intention depends on the particular word in the sentence and different words have different degree of influence on the intention detection, so the basic importance and contribution of each word is estimated by the self-attention mechanism.

We utilize a two-layer multilayer perceptual machine (MLP) and set the activation function to tanh (the last activation function is set to none).

By using the multilayer perceptron, we perform some computations or transformations on the hidden state h_i to obtain an attention function, and the output probability distribution is the attention weight a_i for each word.

The complete calculation is as follows:

$$logit_i^{slot} = W_{slot}\widehat{hd}_i, logit^{intent} = W_{intent}h \quad (2)$$

$$h = MLP\left(\left[\sum_{i=0}^{N-1} softmax\left(MLP\left(\left[\widehat{hd}_i, \overline{hd}\right]\right)\right)\widehat{hd}_i, \overline{hd}\right]\right) \quad (3)$$

where MLP is a two-layer multilayer perceptual machine and the activation function is $tanh$, $softmax$ is performed to obtain attention value, h is the final discourse-level representation, and i is the word index, $W_{slot} \in R^{h_s \times L_s}$ and $W_{intent} \in R^{h_s \times L_i}$ are the mapping matrices, where L_s is the tag size of slot, L_i the tag size of intent, h_s is the hidden size of $BiLSTM$.

3.2.4. Mask Mechanism

K is the number of words in the sentence. T is the total number of iterations.$Tag_i^t \in R^{L_s+L_i}$ is the label distribution of the $i-th$ node in the $t-th$ iteration, where $t \in [0,T], i \in [0,K]$ and Tag_K^t is the label distribution of the intent node, we add the intent as the last node to Tag and apply Mask mechanism to Tag_i^t to recover the original set of labels of the $i-th$ node. Intent nodes have only intent-specific label sets of size L_i, so the mask of an intent node is $[-inf,] * L_s, [0,] * L_i$ like follows:

$$mask_i = \begin{cases} [0,] * L_s, [-inf,] * L_i, if\ i < K \\ [-inf,] * L_s, [0,] * L_i, if\ i = K \end{cases} \quad (4)$$

where L_i is the tag size of intent and L_s is the tag size of slots.

We can obtain the unary potential by following equations:

$$una_i = \begin{cases} [logit_i^{slot}, Zero_{L_i}], if\ i \in [0, K-1] \\ \left[Zero_{L_i}, logit^{intent}\right]. if\ i = K \end{cases} \quad (5)$$

where una_i is the unary potential of the i-th node;$Zero_i$ denotes the zero vector of length l; depending on the mask, Zero vector will be used to fill the original logits.

3.2.5. Graph Construction

First for each word w_i , the k nearest neighbour sets $\mathcal{N}$ are queried from the training set using their corresponding embeddings $\widehat{hd}$, and the L2 Euclidean distance d($\widehat{hd}$, -) is used as the similarity measure. The retrieved nearest neighbor sets $\mathcal{N}$ are represented as (key, value) pairs, where keys denote retrieved similar words and values denote corresponding SL labels. We need to construct a graph from the k nearest neighbours of the query and the input words. Messages are passed from the nearest neighbours to each input word to obtain an aggregated representation.

Then define three types of nodes and different types of edges associated with the nodes:

- a_{input}, which represent the input nodes, i.e. enter the words in the sentence
- $a_{neighbour}$,which represent the neighbour nodes, i.e., the retrieved neighbour terms. In addition to this, the context of the nearest neighbours (thus considered as neighbour nodes) is also included in order to obtain richer contextual information for the retrieved neighbours.
- a_{label},which represent the label nodes. Since the labels of the nearest neighbours provide an important basis for the classification of the input nodes, we would like to pass the influence of the neighbours' labels along the graph to the input nodes The first paragraph after a section or subsection heading should not be indented; subsequent paragraphs should be indented by 5 mm.
- $\Upsilon_{input-input}$,which represent edges from input nodes to input nodes.
- $\Upsilon_{input-neighbour}$, which represent edges between the input nodes and the neighbour nodes.
- $\Upsilon_{neighbour-neighbour}$, which represent edges from neighbour nodes to neighbour nodes
- $\Upsilon_{neighbour-label}$, which represent edges between the neighbour nodes and label nodes

We construct an undirected graph based on the defined nodes and edges, and then use a graph neural network (GNN) to summarize the graph-based information to obtain the final representation of each word to be labelled.

For each edge (s, e, n), the message passed from node s to node n can be expressed as follows:

$$M(s,e,n) = W_{\mathcal{T}(s)}^{v} h_s^{l-1} W_{\phi(e)} \quad (6)$$

where v denotes a collection of nodes v,$W_{\mathcal{T}(s)}^{v} \in \mathbb{R}^{d \times d}$ and $W_{\phi(e)} \in \mathbb{R}^{d \times d}$ are two learnable weight matrices that control the outflow of node s from the node side and the edge side, respectively, d denotes the dimension of a vector.

And we map the source node to a vector $K(s)$ and the target node to a vector $Q(n)$ in order to facilitate the introduction of attentional weights. The attention mechanism can be viewed in Vaswani et al..[9] We can think of the importance of each neighbor of the receiving node n with respect to that relation as an attentional mechanism linking the different locations of a single sequence:

$$K(s) = W_{\mathcal{T}(s)}^{k} h_s^{l-1}, Q(n) = W_{\mathcal{T}(n)}^{q} h_n^{l-1} \quad (7)$$

where $W_{\mathcal{T}(s)}^{k} \in \mathbb{R}^{d \times d}$ and $W_{\mathcal{T}(n)}^{q} \in \mathbb{R}^{d \times d}$ are two learnable matrixes.

When using different types of edges for node connections, we follow Hu et al. to maintain a different edge matrix $W_{\phi(e)} \in \mathbb{R}^{d \times d}$ for each edge type between the points of $K(s)$ and $Q(n)$:

$$P(n,s) = K(s) W_{\phi(e)} Q(n)^T \cdot \frac{\mu\langle \mathcal{T}(s), \phi(e), \mathcal{T}(n)\rangle}{\sqrt{d}}, \quad (8)$$

$$A(s,e,n) = softmax\left(P\left(Q(n), K(s)\right)\right), s \in \mathcal{N}(n), e \in \phi(e) \quad (9)$$

where μ is a learnable matrix representing the contribution of each edge with different relations. [10]

We now have information and attention weights for each edge, and further go to get information and weights for all neighbouring nodes:

$$Aggregate(\cdot) = W_{\mathcal{T}(n)}^{O}\left(\oplus MultiHead(s,e,n)\right), \forall s \in \mathcal{N}(n) \quad (10)$$

$$MultiHead(z,e,n) = Concat\left(head_1 \cdots head_g\right) W^O \quad (11)$$

$$head_i = A(s,e,n) \cdot M(s,e,n) \quad (12)$$

where g is the number of heads of the polytope note, and W^O is a learnable model parameter, $\oplus$ is elementwise addition and $W_{\mathcal{T}(n)}^{O} \in \mathcal{R}^{d \times d}$ is a learnable model parameter used as an activation function like a linear layer.

We define the layer 1 representation of node n as follows:

$$h_n^l = Aggregate\left(A(s,e,n) \cdot M(s,e,n)\right) + h_n^{l-1}, \forall s \in \mathcal{N}(n) \quad (13)$$

where $Aggregate(\cdot)$ denotes the function that aggregates the information passed by the neighbours of node n , $M(s,e,n)$ denotes the information passed by node s to node n along edge e, and $A(s,e,n)$ denotes the edge weights modeling the importance of source node s to target node n.

The aggregated representation of each input word is used as its final representation, passed to the $softmax$ layer to

output the most likely sequence of slots and intent.

4. Experiment and Discussion

This section explores the experimental setup by presenting training details and discussing experimental results.

4.1. Training Details

The $BiLSTM$ layer and $BiGRU$ layer have a consistent number of 100 neurons. The results of the bi-directional layer are imported into the self-attentive layer and then passed into the undirected graph constructed in our model, thus effectively capturing the interdependencies between labels in the output sequence. The middle layer is activated by the $ReLU$ activation function. Furthermore, to offset the risk of overfitting inherent in neural networks, a spatial culling layer with a coefficient of 12% was strategically introduced as a regularization technique before applying the $BiLSTM$ and $BiGRU$ layers.

For the optimization strategy, the Adam optimizer was chosen with a learning rate set to 0.1. In addition, the parameter updates of this model were controlled by the categorical cross-entropy, which was a key driver for the refinement of the model. This comprehensive model was trained over a long period of time in an effort to utilize the computational power of multiple GPUs provided by Kaggle to meet the demands of the sample data.

4.2. Results and Discussion

In our study, we chose F1 score as a measure of slot filling performance and accuracy as a metric of intent detection performance. By combining these two metrics, we are able to comprehensively evaluate the performance of our proposed model and ensure that it achieves satisfactory results in both slot filling and intent detection.

In this study, we focus on comparing our proposed model $BERT-BiLSTM-GNN-SL$ with the classical model BiLSTM.In addition, we also compare these two models with two other benchmark models, $BiLSTM$ and BERT-BiLSTM, which were trained on the full dataset. The relevant comparison data is detailed in Table 1.

Table 1. Comparison of results with published models on the Snips dataset.

Dataset	Model	F1-score(slot)	Accuracy(intent)
Snips Random 15% Subset	$BERT+BiLSTM+GNN-SL$	82.4	87.9
Snips Random 15% Subset	$BiLSTM$	76.7	85.1
Snips	$BiLSTM$	87.8	96.7
Snips	$BERT+BiLSTM+LC-CRF$	96.3	99.2

From the data tables we present, it is clear to observe that our proposed $BERT-BiLSTM-GNN-SL$ exhibits better performance relative to the conventional $BiLSTM$ model in the same sample space. Although in this particular case, the significant reduction in the dataset size leads to a degradation of the model performance, our model has an outstanding advantage when facing the training of small-scale datasets. Not only is it able to significantly reduce the training period, but it is also able to maintain a good level of performance. This is important for applying natural language understanding models to resource-constrained environments.

Overall, our study not only provides an in-depth exploration of improving model performance, but also takes into account the real-world situation of resource constraints during the experimental process. Our proposed model is not only a breakthrough in terms of accuracy, but also has practical applications in terms of resource consumption. This will provide useful guidance and inspiration for tasks in constrained environments in the field of natural language processing

5. Conclusion

This study aims to explore the construction of joint models for intent recognition and slot filling in a constrained resource environment. Our work emphasizes how efficient natural language understanding can be achieved in resource-constrained situations through clever design and innovative approaches. By integrating BERT, $BiLSTM$, graph neural networks, and masking mechanisms, our proposed model overcomes the challenges posed by resource constraints and successfully achieves excellent performance in both intention recognition and slot filling tasks.

Our results show that even when trained on small-scale datasets, our joint model is still able to achieve significant performance gains in resource-constrained environments. This not only confirms the effectiveness of our approach, but also highlights its potential value in real-world applications. This research has practical implications for building efficient natural language understanding models in resource-constrained environments such as mobile devices and embedded systems.

However, we also recognize that there are still many directions worth exploring. Future research can further explore more lightweight model design, finer model fine-tuning, and smarter resource utilization strategies for more comprehensive and efficient natural language understanding. In conclusion, this study provides a solid starting point for joint intent recognition and slot-filling model construction under constrained resources, and provides useful reference and inspiration for research and practical applications in related fields.

References

[1] Jeong, M., & Lee, G. G. (2008). Triangular-chain conditional random fields. IEEE Transactions on Audio, Speech, and Language Processing, 16(7), 1287–1302.

[2] Guo, D., Tur, G., Yih, W. T., & Zweig, G. (2014). Joint semantic utterance classification and slot filling with recursive neural networks. Proceedings of the IEEE Spoken Language Technology Workshop (SLT'14, pp. 554–559.

[3] Raedt, L. D., Hammer, B., Hitzler, P., & Maass, W. (2008). Recurrent Neural Networks - Models, Capacities, and Applications. In: Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI). Schloss Dagstuhl, Germany. 08041

[4] Luo, X., Zhou, W., Wang, W., Zhu, Y., & Deng, J. (2018). Attention-based relation extraction with bidirectional gated

recurrent unit and highway network in the analysis of geological data. IEEE Access, 6, 5705–5715.

[5] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780.

[6] Tang, H., Ji, D., & Zhou, Q. (2020). End-to-end masked graph-based CRF for joint slot filling and intent detection. Neurocomputing, 413, 348–359.

[7] Wang, S., et al. (2023) GNN-SL: Sequence Labeling Based on Nearest Examples via GNN. In: Findings of the Association for Computational linguistics (ACL 2023). Toronto. 12679-12692.

[8] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 1: 4171-4186.

[9] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

[10] Hu, Z., Dong, Y., Wang, K., & Sun, Y. (2020). Heterogeneous graph transformer. Proceedings of the Web Conference 2020. 2704–2710.