SLOT FILLING

Bolunzhang MingboDai Chaofeng
521030910414 519030910364 521030910413

ABSTRACT

The rapid progress in spoken language understanding (SLU) systems over the past few decades has led to significant advancements in intent detection (ID) and slot filling (SF) tasks. ID aims to comprehend the user's intended meaning, while SF focuses on extracting semantic information from sentences using sequence labeling techniques.

This paper primarily focuses on slot filling, driven by the characteristics of the dataset used. Initially, we explored a baseline model made by BiLSTM and identified the limitations of the baseline approach. Subsequently, we used modifications to enhance its accuracy and leveraged ctran to adapt it to our specific dataset. Through qualitative analysis, we observed that ctran outperformed the baseline model.

By addressing the shortcomings of the baseline and introducing ctran, this report demonstrates the effectiveness of adapting existing approaches to improve slot filling in SLU systems. The findings highlight the significance of selecting suitable techniques based on dataset characteristics and provide insights into enhancing the overall performance of SLU systems.

***K*eywords** Noise Reduction · Historical Dialogues · CTRAN

1 Introduction

The research on spoken language understanding (SLU) system has progressed extremely fast during the past decades. Two important tasks in an SLU system are intent detection and slot filling. The first is to understand the intent that the user has in mind, and the second is to extract the semantic information from the sentence. ID can be defined as a classification problem,and SF can be considered as a sequence labeling problem.

In this report, we focus on Slot-Fitting,which is due to the dataset. First, we ran the baseline and made various modifications based on the suggestions to improve its accuracy. Then, we analyzed the shortcomings of the baseline and adapted ctran for our dataset. Qualitative analysis revealed that ctran outperformed the baseline, and we discussed the reasons for its superior performance.

2 Related Work

2.1 baseline

The sequence annotation method based on BiLSTM[1] performs well in semantic understanding tasks, especially suitable for handling tasks with short texts and local context dependencies. However, for tasks with long-range dependencies and global optimization, it may be necessary to combine other technologies or models to further improve performance. For example, due to BiLSTM's prediction based on context rather than global optimization, the model overly relies on the preceding or following labels when predicting the current label, while ignoring the global information of the entire sequence. And BiLSTM can capture local context dependencies, its modeling ability is limited for dependencies over long distances. This may result in some complex semantic understanding tasks not performing well when using BiLSTM for sequence annotation.

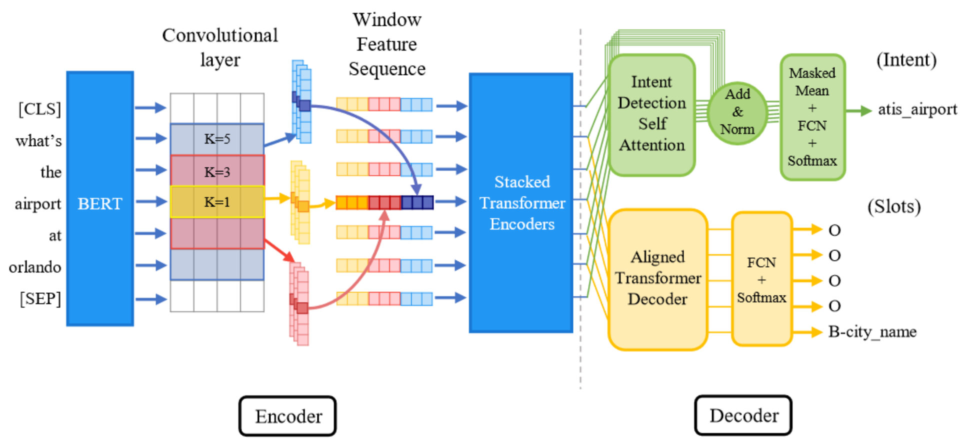

Figure 1: CTRAN

2.2 Traditional Models

Traditional models mostly used pre-trained fixed vector embeddings such as Word2Vec [2] for the word representation.Then the most straight-forward way is using single RNN model[3] generating multiple semanctic tags sequentially by reading in each word one by one.This approach has a constrain that the number of slot tags generated should be the same as that of words in an utterance.Also,they can use other vector embeddings. For example, CoBiC[4] uses CNN to process the embeddings of the input tokens . The output of the CNN is fed to a bi-directional LSTM, and then a conditional random field (CRF) layer is used to generate the output labels.

Another way is by using an encoder-decoder model containing two RNN models as an encoder for input and a decoder for output[5]. The advantage of doing this is that it gives the system capability of matching an input utterance and output slot tags with different lengths without the need of alignment.

In recent years, pre-trained language models have become progressively adopted.NLU models either use the language model as word embeddings or use the structure of the language model as their encoder and only propose a decoder.For example, Chen et al. [6] used BERT as an encoder, harnessed [CLS] token with a softmax layer for ID and rest of the token outputs with a softmax for generation of target slots.

2.3 CTRAN

CTRAN[7] is a encoder-decoder architecture.CTRAN comprises three main components: A shared encoder, an SF decoder, and an ID decoder. SF and ID tasks use the same encoder, meaning both tasks use the vector produced by the encoder, and the subsequent losses are summed before calculating the gradients.However we don't need to use the ID decoder in our task.Fig1 are the abstract of CTRAN:

3 Improvements based on the baseline

3.1 Dataset Analysis

We tabulated some of the distributions of the datasets, and the problems are as follows:

1.The data is unevenly distributed. For example, the action of deny accounts for only about 0.1% (5/5119) of the dataset. This can lead to overfitting of the trained model.

2.Some speech recognition content may have misidentification and omission, and it is difficult to directly match, replace or complete based on the existing vocabulary data. For example, manual_transcript :

3.Background noise is severe. Due to the limited environment in the car, the data is noisy and may be interfered with by the navigation prompt sound or the passenger's communication sound, corresponding to the annotation in the data set (side) (dialect) (robot).

Slot fitting

Table 1: Baseline Model

Dataset	**Dev acc**	**Dev fscore**$p/r/f$
asr_1best	71.28	81.26/74.14/77.54
manual_transcript	**93.18**	**94.12/95.10/94.61**

3.2 Problems with the baseline model

The problem with the baseline model is that it ignores historical information. The model inputs independent instruction statements for understanding, ignoring the previous content, which makes it difficult for pronouns to directly fill in slot values in some examples.

Intention is indirectly obtained, and some data contains information that cannot be directly obtained from the text and cannot be directly filled with slot values. Instead, it needs to be extracted by combining semantics.

The performance improvement brought by a good dataset is enormous. By simply modifying the function loaddataset and using manualtranscript dataset for training, an accuracy rate of over 90% can be achieved directly. In the subsequent experiments, we only used dataset asr1best.

3.3 Noise Reduction

In the process of converting speech into written text, recognition errors often occur due to issues such as pronunciation, dialects, homophones, and so on. This phenomenon is particularly common for certain specialized geographical terms, as the lack of contextual logic support exacerbates the problem. To address this, we introduce character edit distance, specifically the Levenshtein distance[8], to measure the similarity between two strings. For inputs with noise, we iterate through a vocabulary list to find the word with the closest Levenshtein distance to the target word and use it as a replacement.

Algorithm 1 Noise Reduction Algorithm

```
1: Input: Noisy text, Word dictionary
2: Output: Cleaned text
3: Procedure: ReduceNoise
4: $noisyText$ ← ReceiveNoisyInput()
5: $wordDictionary$ ← LoadWordDictionary()
6: for each word $w$ in $noisyText$ do
7:     if $w$ not in $wordDictionary$ then
8:         $closestWord$ ← FindClosestWord($w$, $wordDictionary$)
9:         $noisyText$ ← ReplaceWord($noisyText$, $w$, $closestWord$)
10:    end if
11: end for
12: return $noisyText$
```

After applying the denoising algorithm, our test results have shown a significant improvement.

Table 2: Lev Denoising algorithm

Denoise	**Dev acc**	**Dev fscore(p/r/f)**
Baseline	71.28	81.26/74.14/77.54
Lev Denoise	**80.00**	**92.15/80.81/86.11**

3.4 Using Historical Dialogues

Due to the fact that each sample in the existing baseline only contains the current conversation sentence, without considering the content previously mentioned by the user, it may result in a lack of key information. For this reason, we attempt to add historical dialogues as a reference when inputting the model.

Specifically, there are some consecutive sentences in the dataset that are placed in the same list and labeled with the "utt_id" tag to indicate their order of occurrence. During training, the incoming data not only includes the current sentence, but also all previous sentences that have appeared before. After merging them, they are used as a complete sentence for prediction. After the prediction is completed, only the prediction result corresponding to the current sentence is retained.

However, the structure of the baseline model is too simple, and when the input string becomes longer, it cannot effectively extract historical information. The information from historical dialogues has caused interference with the current dialogue, and the model mistakenly regards historical information as the core semantics, ignoring the semantics of the current dialogue.

Table 3: History

Algorithm	Dev acc	Dev fscore(p/r/f)
Baseline	71.28	81.26/74.14/77.54
History	**68.62**	**73.80/71.34/72.21**

3.5 Incorporating Pre-trained Models

We incorporated pre-trained models into the baseline experiment, replacing the Word2Vec[2] word embeddings. Specifically, we utilized four pre-trained models for word embeddings: Bert-base-Chinese (BBC), Chinesebert-wwm-ext (CBWE), Chinese-electra-180g-basediscriminator (CEBD), and gpt2-base-Chinese (GBC). The experimental comparison results are as follows, with all algorithms applied with lev denoising.

Table 4: Pre-trained Models

Pre-trained Model	Dev acc	Dev fscore(p/r/f)
word2vec (baseline)	80.00	92.15/80.81/86.11
CBWE	83.44	90.61/82.36/87.74
GBC	82.01	88.61/83.11/86.39
CEBD	81.23	88.01/82.83/84.39
BBC	**84.47**	**92.91/85.56/89.09**

4 CTRAN adaption on slu

We adopt CTRAN on this work. We don't need to build ID,so we just need adapt it partially. Also we run it on asr_ best.Here are our results.

4.1 result

Table 5: CTRAN

Model	Dev acc	Dev fscore(p/r/f)
Baseline	71.28	81.26/74.14/77.54
CTran	**92.5**	**98.83/ 99.64/99.23**

4.2 Analysis

We can find that CTRAN is much better than baseline,here are the possibile reasons:

- pre-trained model We use pre-trained model rather than word2vec.Word2vec is a static word vector pre-training model, and the word vector is fixed, which cannot solve the problem of polysemous words and cannot consider the expected global information
- encoder-decoder RNNs cannot capture the deep bidirectional meaning of a sentence since their output vector is a concatenation of two unidirectional RNNs,Transformer addressed this problem with self-attention and positional embeddings.So it performs well.

Slot fitting

5 Optimal Solution

Finally, we comprehensively considered all the above optimization strategies:

- **Dataset**:manual transcript
- **Algorithm1**:Lev Noise Reduction Algorithm
- **Algorithm2**:Using Historical Dialogues
- **Pre-trained Models**:Bert-base-Chinese (BBC)
- **Model**:CTRAN adaption

We selected the **manual transcript dataset**, applied the **lev denoising algorithm** and the **pre-trained model BBC**, and obtained the optimal solution

Table 6: Optimal Solution

Method	Dev acc	Dev fscore(p/r/f)
Baseline	71.28	81.26/74.14/77.54
Manual transcript+Lev+BBC	**94.25**	**97.57/94.44/95.98**

6 Conclusion

In conclusion, we focused on slot filling in spoken language understanding (SLU) systems and presented an analysis of the baseline model's shortcomings. Through various modifications and adaptations using ctran, we significantly improved the accuracy of slot filling on our dataset.

7 Assignment of Responsibilities

BolunZhang:Pre-training model addition and experimentation ,Levenshtein denoising algorithm implementation and experimentation.

Chaofeng:Literature research,Ctran adaption and evalution, definition of problem and search related works.

MingboDai:Dataset analysis and baseline model problem exploration, Historical model implementation and experimentation.

References

[1] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging, 2015.

[2] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[3] Spoken language understanding and interaction: Machine learning for human-like conversational systems - ScienceDirect.

[4] Bamba Kane, Fabio Rossi, Ophélie Guinaudeau, Valeria Chiesa, Ilhem Quénel, and Stéphane Chau. Joint Intent Detection and Slot Filling via CNN-LSTM-CRF. In *2020 6th IEEE Congress on Information Science and Technology (CiSt)*, pages 342–347.

[5] Bing Liu and Ian Lane. Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling.

[6] [1902.10909] BERT for Joint Intent Classification and Slot Filling.

[7] Mehrdad Rafiepour and Javad Salimi Sartakhti. CTRAN: CNN-Transformer-based network for natural language understanding. 126:107013.

[8] Li Yujian and Liu Bo. A normalized levenshtein distance metric. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1091–1095, 2007.