SSN: A Spatial Semantic Network for High-accuracy, Stable and Generalized Intent Detection

Zhi, Zeng
Department of Computer Science
City University of Hong Kong
Hong Kong
zhizeng8-c@my.cityu.edu.hk

Chi-Yin Chow
Department of Computer Science
City University of Hong Kong
Hong Kong
chiychow@cityu.edu.hk

Jia-Dong Zhang
Department of Computer Science
City University of Hong Kong
Hong Kong
jzhang26@cityu.edu.hk

***Abstract*—Nowadays, deep learning-based chatbot technologies have attracted the attention of many industries. Natural language understanding which consists of intent detection and slot filling is the first step in building a chatbot. Intent detection alone can be used to build a chatbot to answer frequently asked questions. However, there are still three major challenges in intent detection, namely, accuracy, stability and generalization ability. To overcome these three challenges, this paper proposes a new spatial semantic network (SSN) for intent detection method based on sentence embedding and support vector machine. Our SSN method consists of a frontend mapping a sentence to a dense vector in high dimensional space and a backend classifying the points into different intents. We conducted experiments on two popular real-world datasets. Experimental results show that our SSN method outperforms the state-of-the-art methods in terms of accuracy, stability as well as the ability to generalize on small training set.**

***Index Terms*—Intent detection, chatbot, sentence embedding, support vector machine**

I. Introduction

Chatbot is a computer program which can talk to a human-being by voice or text. It can simulate the dialog between human-beings, and even aims at passing the Turing test. Chatbot can be used in practical scenarios, such as customer services or information acquisition. Some chatbot has natural language processing system, but most simple systems can only select keywords from the input and then find the most suitable response from the database. Nowadays, chatbot is a part of the virtual assistant, e.g. google smart assistant, and can connect with the applications, websites, instance message platforms of different organizations. Non-assistance applications include chatroom for entertainment, research, product promotion, and social chatbot.

To build a chatbot, many attempts have been made. The most simple kind of chatbot is rule-based. It is easy to implement, but is hard to satisfy the diverse need of the customers because of the richness of human natural language. There are also chatbots built by end to end approach, similar to the way neural machine translation was done. However, such a kind of chatbot lacks consistency in answering questions. It may have different answers when asked how old are you and what is your age. To improve the quality of chatbot, the pipeline method consisting of several different modules including speech recognition, natural language understanding, dialogue management, knowledge provider as well as natural language generation are developed.

Natural language understanding is a critical step in building a chatbot by the pipeline method. Because only when the chatbot is able to understand what the user wants to do, the chatbot can return a reasonable response. The natural language understanding module consists of two parts, which are intent detection and slot filling. Intent detection is the process of understanding what the user wants to do, while slot filling means the process of filling all necessary information, in order to transform customer intent into explicit machine instruction.

Nowadays, chatbot based on deep learning techniques has attracted the attention of many industries [1] [2] [3]. Intent detection is an important step in building a chatbot. With intent detection alone, FAQ-bot which is used in answering repetitive and common customer service questions [4]. Techniques in doing intent detection can be categorized into non-deep learning methods and deep learning method. Non-deep learning method uses hand-crafted features like Term Frequency Inverse Document Frequency (TFIDF) while deep learning method trains neural networks to learn features on their own.

Most existing works used deep learning models. Liu and Lane, 2016, showed the ability of sequence to sequence deep learning model in the field of natural language understanding [5]. Hakkani-Tur et al., 2016, presented an encoder-decoder architecture and an attention based architecture [6]. Goo et al., 2018, added a slot gate module in the attention based architecture proposed by Liu and Lane in order to model the explicit relationships between slots and intent vectors [7].

This paper mainly focuses on three research challenges on intent detection for chatbot. (1) Improving the accuracy of intent detection. Previous method has reached relatively high accuracy, but there is still space to improve. (2) Making the intent detection system perform more stable under different testing sets. The weakness of previous approaches is that the

978-1-7281-4117-6/19/$31.00 ©2019 IEEE
DOI 10.1109/DataCom.2019.00010

accuracy fluctuates even if the training set and testing set are exactly the same when run multiple times. (3) Strengthening the ability to generalize. In practice, people hope the system can perform well when the training set is as small as possible. Our proposed method tries to address these challenges.

To overcome the challenges described above, we propose a new Spatial Semantic Network (SSN) for intention detection based on sentence embedding and support vector machine. It is designed for high accuracy, stable and well-generalized intent detection system. At training time, in the frontend module, a sentence embedding encoder is pre-trained and used to encode an utterance into a dense vector in high dimensional space; in the backend module, a support vector machine is trained by these dense vectors together with their corresponding labels. At testing time, in the frontend, the new utterances are encoded by the pre-trained sentence embedding encoder; then in the backend, these dense vectors are classified into different intents by the trained support vector machine classifier. Our main contributions can be summarized as follows:

- We designed a SSN that not only makes use of word embedding but also sentence embedding, which enables our system to understand the meaning of a sentence as a whole better.
- We conducted several experiments on two different popular datasets using our proposed method. Experimental results shows that our proposed method obtained higher intent detection accuracy.
- Our SSN performs more stable than previous methods. When run repeatedly, previous methods achieve different accuracy even if the training set and testing set are the same. By contrast, SSN generates the same accuracy when run several times. Furthermore, SSN also achieves lower standard deviation when tested on different testing set.
- Our SSN is more capable of generalizing on small size training set. The accuracy of previous methods will drop considerably if the size of training set decreases significantly. On the contrary, SSN can keep a high accuracy compared with other methods even if the training set shrinks, which means that SSN has greater ability to generalize.

The rest of the paper is organized as follows. Section 2 introduces related work. Section 3 presents the problem definition. Section 4 elaborates the limitations of previous method. Section 5 shows the details of our proposed method. Section 6 evaluates the performance of our method compared with other state-of-the-art methods. Finally, we conclude this paper in Section 7.

II. Related work

This section briefly highlights the related work about intent detection, word embedding and sentence embedding.

A. Non-deep learning methods

TFIDF-LR and TFIDF-SVM [8] are two kinds of non-deep learning methods that are used to do intent detection. TFIDF [9] stands for term frequency inverse document frequency. It is a statistical method used to measure how important a word is in a collection of documents by considering the frequency of the word and document. The advantage of TFIDF is that it can filter out some unimportant words in the sentence and keep those words that are important. Term frequency measures how frequently a word appears in a document. Inverse document frequency decreases the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely. After using TFIDF to represent the utterances, the sentences are classified using logistic regression or support vector machine.

B. Deep learning methods

Hakkani-Tur et al. proposed Multi-Domain Joint Semantic Frame Parsing using Bi-directional RNN-LSTM [5]. It shows the ability of sequence to sequence deep learning model in the field of natural language understanding. Attention-based recurrent neural network models for joint intent detection and slot filling was proposed by Liu and Lane [6]. In this paper, they presented an encoder decoder architecture and an attention based architecture. Later on, Goo et al. proposed Slot-Gated Modeling for Joint Slot Filling and Intent Prediction [7]. In this paper, they add a slot gate module in the attention based architecture proposed by Liu and Lane in order to model the explicit relationships between slots and intent vectors.

C. Word embedding

Before the existence of word embedding, the word appears in a sentence are encoded by assigning an integer or by one-hot encoding. Such representation can be used in some simple NLP task, but it lose the meaning of the words. By contrast, word embedding representation embed the sparse representation into a low dimensional dense representation and keeps the meaning of words, which means semantically similar words will be located near each other. Word2vec [10] is one of the famous word embedding techniques. In addition, there exists more advanced techniques including GloVE [11] and Fasttext [12] [13] [14]. However, word embedding alone is not enough to figure out the intent of an utterance so we also need sentence embedding which will be introduced later.

D. Sentence embedding

Sentence embedding is a dense vector in high dimensional space generated form a sentence by an encoder. Sentences with similar semantic meaning will be located near each other in the sentence embedding representation. There are many kinds of existing sentence embedding techniques, including standard recurrent encoders with either Long Short-Term Memory (LSTM) or Gated Recurrent Units (GRU), concatenation of last hidden states of forward and backward GRU, Bi-directional LSTMs (BiLSTM) with either mean or max pooling, self-attentive network and hierarchical convolutional networks [15].

1) LSTM and GRU: The simplest kind of sentence embedding technique is using LSTM [16] or GRU [17] units. Given the word sequence containing several words as input, the hidden states represented by several LSTM or GRU hidden states are generated with rightward recurrent LSTM operation or rightward recurrent GRU operation. The sentence embedding is represented by the last hidden state of LSTM or GRU unit. In addition, a bidirectional LSTM or GRU model can also be used by concatenate the last hidden state of the forward LSTM/GRU and the first hidden state of the backward LSTM/GRU as the sentence embedding vector.

2) BiLSTM with mean/max pooling: The BiLSTM with mean/max pooling architecure consists of a bidirectional LSTM layer and a mean-pooling or max-pooling layer. Given a word sequence as in put, the forward LSTM generates a sequence of hidden states, the backward LSTM generates another sequence of hidden states, then these two sequences of hidden states are concatenated componentwisely [18]. Then the concatenated hidden states are processed by max-pooling (selecting the maximum value of each dimension) or average-pooling (calculate the average value of each dimension). The sentence embedding is represented by the processed fixed-size vector. We can notice that in this way more information of the sentence and more hidden states information are preserved in the sentence embedding vector.

3) Self-attentive network: The self-attentive network contains bidirectional LSTM and attention mechanism. Given a word sequence as input, the concatenated hidden states represented by several LSTM hidden states are generated. These hidden states then go through an affine transformation and hyperbolic tangent activation function to generate the keys of each LSTM unit. Then the similarity between the keys and context vector is calculated. Finally, sentence embedding vector is a weighted combination of the similarity and hidden state [19] [20].

4) Hierarchical ConvNet: Inspired by the convolutional architecture termed AdaSent, researchers from Facebook introduced an architecture with four convolutional layers. The representation of each hierarchy is computed by max-pooling operation at each layer. Finally, the sentence embedding vector is represented by a linear concatenation of each hierarchical representation from the four layers [21]. In this way, the hierarchical abstraction can be captured from the input sentence by the network.

E. Limitations of previous methods

The first limitation of previous method is that the intent detection accuracy can still be higher. The second limitation is that the accuracy of previous methods is sometimes better and sometimes worse, which means that it is not stable enough. The third limitation is that when we cannot collect massive training data, the accuracy of previous methods tend to decline in a big scale. Therefore, we propose a new method which can achieve higher intention detection accuracy more stability as well as greater ability to generalize on small size training set.

III. Proposed method

This section is organized as follows. Firstly, the problem definition is stated. After that, the idea behind the method is introduced. Then the overall architecture of the proposed method is presented. Later on, the details of the frontend and backend are described.

A. Problem definition

The intent detection problem can be regarded as an utterance classification problem. In the training set, each utterance is associated with a label called intent. Each sample in the training set can be described as (x, y), where x represents the utterance and y represents the intent. Intent y is one of the elements in the intent category set, which can be described as $Y = \{y_1, y_2, \ldots, y_n\}$. The task is to find the appropriate intent label given a new utterance from the testing set.

B. Idea of the method

The idea of our method contains two steps. The first step is to map an utterance to a dense vector in high dimensional space. Such vector is called sentence embedding. Sentence embedding enables sentences having similar semantic meaning to be located near each other, just like word embedding enables words having similar semantic meaning to be located near each other. The limitation of word embedding is that although it knows "chicken", "hen" and "rooster" are close to each other, it does not know "male chicken" equals "rooster" or "female chicken" equals "hen". As a result, the meaning of a sentence composed of several words cannot be described solely by word embedding. Fortunately, due to the progress in sentence embedding research, we can now do the mapping form sentences to dense vectors in high dimensional space through deep learning. The sentence embedding is trained on vast dataset and not restricted to the intent detection training dataset which is small comparatively. As a result, our proposed method is able to generalize better than previous method when the utterances appear in the testing set have not been shown in the training set. The second step is do the mapping from dense vectors to intent labels. Essentially, this is dividing the high dimensional space into several subspaces and draw boundaries between them. As we know, support vector machine is such a boundary divider that can maximize the margin between points from different classes.

C. Architecture of the method

The architecture of our model consists of two parts, the frontend and the backend. The frontend is used for sentence embedding and the backend is used for intent classification. The overall architecture of the proposed method with intermediate computing process is depicted in figure 1. The details of the frontend will be introduced in subsection III-D and the details of the backend will be introduced in subsection III-E.

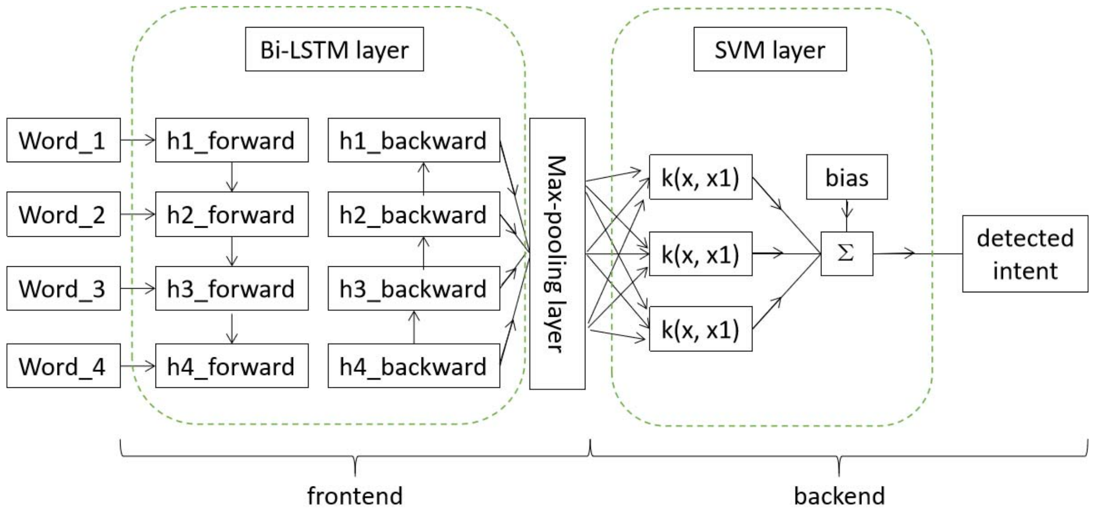

Fig. 1: An overview of the architecture of our SSN.

D. Frontend for sentence embedding

There are many existing neural networks that can be used to encode sentences into fixed size dense vectors in high dimensional space. Such neural networks includes standard recurrent encoders with either Long Short-Term Memory (LSTM) or Gated Recurrent Units (GRU), concatenation of last hidden states of forward and backward GRU, Bi-directional LSTMs (BiLSTM) with either mean or max pooling, self-attentive network and hierarchical convolutional networks. Researchers from Facebook have compared the performances of these seven kinds of sentence encoders and it turns out that the Bi-directional LSTMs (BiLSTM) with max pooling has the best performance. Given a sequence of words $(w_1, w_2, ..., w_T)$ as input, the bidirectional LSTM computes its hidden states both forwards and backwards. The forward and backward results are then concatenated. The formulas are as follow.

$$\overrightarrow{h_t} = \overrightarrow{LSTM_t}(w_1, ..., w_T) \tag{1}$$

$$\overleftarrow{h_t} = \overleftarrow{LSTM_t}(w_1, ..., w_T) \tag{2}$$

$$h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}] \tag{3}$$

Researchers from Facebook have compared the performance of combining the concatenated results by max-pooling (selecting the maximum value over each dimension of the hidden units) and mean-pooling (considering the average of the representations). It turns out that max-pooling has better performance. Therefore, in our proposed method, we choose Bi-directional LSTMs (BiLSTM) with max pooling as our frontend sentence encoder.

E. Backend for intent classification

After going through the frontend, the intent detection problem has been transformed into a point classification problem in high dimensional space. There are many machine learning techniques that can be used to classify points in high dimensional space, including logistic regression, fully connected neural network and support vector machine [22]. After experiments, it turns out that the support vector machine is the best choice because of its few hyper-parameters, stable and accurate performance, as well as good interpretability. The support vector machine can draw a boundary between points belonging to two different classes in high dimensional space such that the gap or margin is maximized by using kernel trick. The binary classification can also be extended to multi-class classification problem. What's more, even if the labels are unknown, we can do unsupervised learning by a variant of support vector machine called the support vector clustering.

F. Training procedure

The training of the frontend and backend are done one after another. Firstly, the frontend is trained on the SNLI (Stanford Natural Language Inference) dataset. The input of the dataset is a sentence pair containing a premise sentence and a hypothesis sentence, the label of the sentence is the relationship of the two sentences, which has three categories including entailment, contradiction and neutral. The two sentences are fed into two identical frontend encoders, the output (u, v) are combined by $(u, v, |u - v|, u * v)$, then fed into a fully-connected layer,producing a predicted category. In the concatenation, u and v are necessary information, while $|u-v|$ and $u*v$ corresponds to the euclidean distance and cosine distance between the two vectors. The loss is computed using the classic cross entropy method and optimized by using stochastic gradient descent. The model has been trained by researchers from Facebook, so we can make use of the pre-trained model. Secondly, we use to well-trained sentence encoder to encode user utterances from ATIS or Snips dataset and obtained many sentence embedding vectors in high dimensional space. We use these vectors and their intent labels to train a support vector machine with linear kernel. After training, the two components, frontend and backend, are combined together, with testing user utterances as input, to generate intent labels as output.

Algorithm 1: SSN training algorithm

```
Input: training data including SNLI dataset and ATIS
       dataset, all hyper-parameters;
Output: trained SNN model;
initialization: all training parameters $\Theta_1$ in the frontend
  and all training parameters $\Theta_2$ in the backend;
/* training the frontend                              */
for each epoch in SNLI dataset do
    for each batch do
        encode sentence pair by untrained BiLSTM
          max-pooling encoder;
        combine two vectors by $(u, v, |u - v|, u * v)$;
        feed into fully connected neural network;
        optimize $\Theta_1$ by minimizing the cross entropy
          loss;
    end
end
/* encoding the sentences into
   embedding vectors                                  */
for each utterance in ATIS dataset do
    encode user utterance by trained BiLSTM
      max-pooling encoder;
end
/* training the backend                               */
for each vector in encoded vector set do
    for each batch do
        compute the loss by
          $Loss = R(w) + C\Sigma max(0, 1 - y_i(wx_i + b))$;
        optimize $\Theta_2$ by minimizing loss;
    end
end
```

In this algorithm, Θ_1 is all the training parameters in the frontend, Θ_2 Θ_2 is all the training parameters in the backend, u is the sentence embedding vector of the first sentence in the sentence pair while v is the sentence embedding vector of the second sentence in the sentence pair. $Loss = R(w) + C\Sigma max(0, 1 - y_i(wx_i + b))$ is the loss function of SVM, where $R(w)$ is the regularizer, C is the penalty parameter for the error term, and $\Sigma max(0, 1 - y_i(wx_i + b))$ is the hinge loss. Hinge loss means that if the class is correctly predicted with a margin, the loss will be zero; otherwise the loss grows linearly.

IV. Experiments

A. Experiment settings

1) Datasets: In this study, we use two popular datasets, including ATIS and Snips[1]. The ATIS dataset contains the utterances about making flight reservations. The training set contains 4,478 utterances and the testing set contains 893 utterances. There are 120 slot labels and 21 intent types in the dataset. Another dataset called Snips is collected from the Snips personal voice assistant, where the number of samples for each intent is approximately the same. The training set contains 13,084 utterances and the testing set contains 700 utterances. There are 72 slot labels and 7 intent types in the dataset. The statistical information of the two datasets are summarized in table 1.

[1]The datasets used in this study can be downloaded from https://github.com/MiuLab/SlotGated-SLU/tree/master/data

TABLE I: Statistics of the datasets

	ATIS	Snips
Vocabulary size	722	11,241
Number of intents	21	7
Training set size	4,478	13,084
Testing set size	893	700

2) Evaluation metrics: The performance of the intent detection system is evaluated by the accuracy and stability. Accuracy is defined by the formula

$$accuracy = \frac{M}{N},$$

where M is the correctly detected intents while N is the total number of actual intents. Stability is defined by the standard deviation of the accuracy on different subsets of the testing set. The formula is

$$SD = \sqrt{\frac{\Sigma_{i=1}^{n}(x_i - \bar{x})^2}{n-1}},$$

where n is the number of data points, x_i represents each of the values of the data, and $\bar{x}$ is the mean of x_i.

3) Hyper-parameter setting: For the frontend, the batch size is set to 64, the word embedding dimension is set to 300, the encoder lstm dimension is set to 2048, and the dropout rate is set to zero. For the backend, the kernel is set to linear, parameter C is set to 1, and other parameters are set to default.

B. Compared models

We compared our proposed method with the following models.

- Joint Seq. [5]: a model built based on LSTM and can capture the sequential information of the sentence.
- Atten.-Based [6]: a model making use of attention mechanism, thus it is able to focus on the key words of the sentence to do classification.
- Slot-Gated (Full Atten.) [7]: a model utilizing slot gated mechanism, a way to model the explicit relationship between slots and intent vectors.
- Slot-Gated (Intent Atten.) [7]: a variant of the third model, which works better on simpler natural language understanding task.

C. Overall comparison

Our proposed methods are called Spatial Semantic Network (SSN) because they transform the semantic classification problem to the spatial classification problem. From the experiment results, we can see that our proposed methods outperform previous methods not only in the ATIS dataset, but also in the Snips dataset. Therefore, it shows good generalization ability.

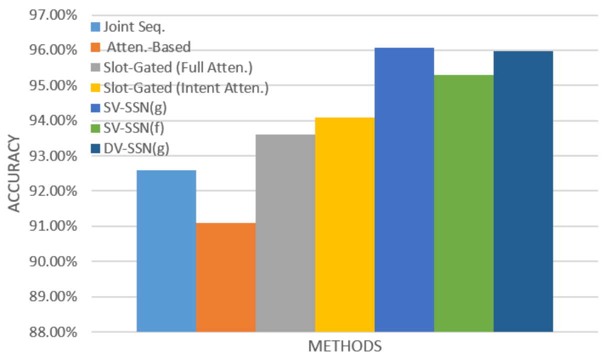

Fig. 2: Comparison of each method on ATIS dataset

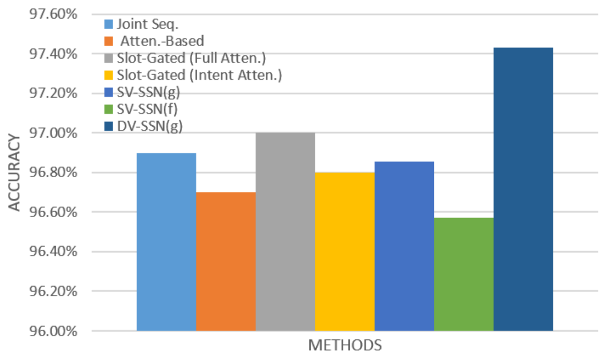

Fig. 3: Comparison of each method on Snips dataset

D. Comparison of variants

There are three different variants of our proposed method (SSN). Version I is called Static Vocabulary Spatial Semantic Network with GloVe (SV-SSN(g)), because it uses GloVe as word embedding and pick the most frequent 10000 words in the dictionary as vocabulary. Version II is called Static Vocabulary Spatial Semantic Network with Fasttext (SV-SSN(f)), because it uses Fasttext as word embedding and pick the most frequent 10000 words in the dictionary as vocabulary. Version III is called Dynamic Vocabulary Spatial Semantic Network with GloVe (DV-SSN(g)), because it uses GloVe as word embedding and pick the words from the sentences as vocabulary. The experiment results shows that GloVe is more suitable to be used as word embedding than Fasttext and having a higher vocabulary coverage rate achieved by dynamic vocabulary will increase and stabilize the performance of our system.

TABLE II: Overall comparison

Methods	**Accuracy in ATIS dataset**	**Accuracy in Snips dataset**
Joint Seq.	92.6%	96.9%
Atten.-Based	91.1%	96.7%
Slot-Gated (Full Atten.)	93.6%	97.0%
Slot-Gated (Intent Atten.)	94.1%	96.8%
Spatial Semantic Network (SSN)	**95.97%**	**97.43%**

TABLE III: Comparison of variants

Methods	Accuracy in ATIS dataset	Accuracy in Snips dataset
SV-SSN(g)	96.08%	96.86%
SV-SSN(f)	95.30%	96.57%
DV-SSN(g)	95.97%	97.43%

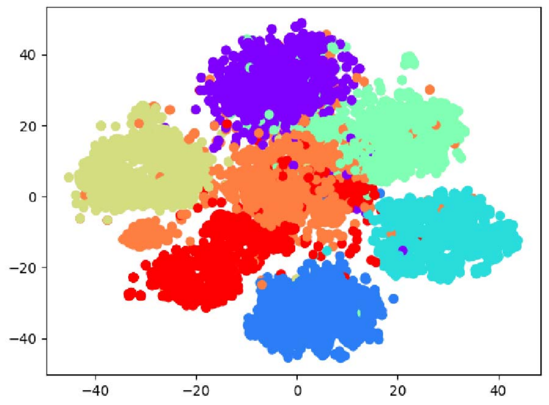

Fig. 4: Sentence embedding distribution of training set

E. Visualization and interpretation

To have a better understanding of how and why our proposed method can achieve such a high accuracy in the intent detection task, we visualize some intermediate computing process. Figure 4 shows the sentence embedding distribution of training set, processed by t-SNE [23]. We can see that sentences labeled with the same intent are clustered together, making it easier for the back end to classify the sentences in the testing set. Figure 5 shows the sentence embedding distribution of training set and testing set. The alpha value of the training points' color is set to 0.25 while the alpha value of the testing points' color is set to 1. We can see that most of the sentences in the testing set falls into the correct category of the region defined by the training set data. There are some exceptions, but parts of it due to the information loss of the t-SNE processing. Figure 6 shows the confusion matrix of the testing set after experiment. It shows the error distribution of the intent detection system.

F. Influence of training set size

The size of training set and the size of testing set can influence the performance of an intent detection system, because deep learning systems are built based on big data. Usually, people hope the system can perform well on a testing set as large as possible while requiring a training set as small as possible, that is a good ability of generalization. This section compares the performance difference of different methods on

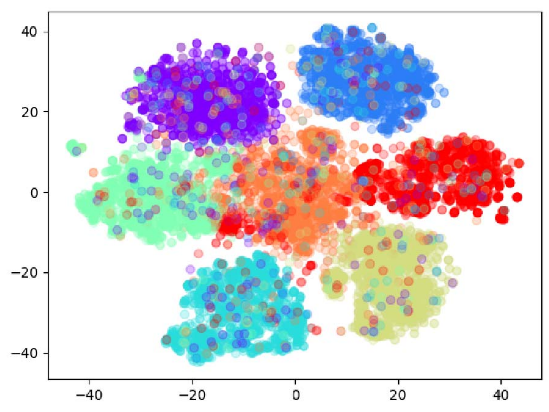

Fig. 5: Sentence embedding distribution of training and testing set

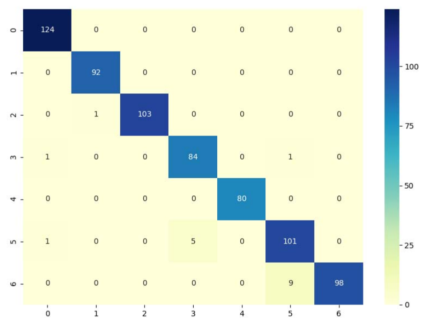

Fig. 6: Confusion matrix of testing set, the x-axis is the actual intent class while the y-axis is the detected intent class

different size of training set and testing set, which shows their different ability to generalize.

In this experiment, training set of 25%, 50%, 75% and 100% of the size of the original training set are used for training different models. The performance difference are shown in the line chart below. It shows that the abilities to generalize are almost the same with little differences on Snips dataset. However, our proposed method apparently has greater capability to generalize on ATIS dataset, because it can keep an accuracy higher than the compared methods even when the size of the training set decreases significantly.

G. *Stability analysis*

The stability of an intent detection is very important, because the testing result of a deep learning model can be different even if the training set and the testing set are exactly the same when the model is run for several times. So we divide the testing set into four parts with thee same size and test different methods for four times. Their performance difference are shown in the line chart and the standard deviations of the accuracy of different methods are compared in the histogram. It shows that our proposed method has the lowest standard deviation compared with other methods on two different datasets. It means that the stability of the proposed method outperforms previous methods.

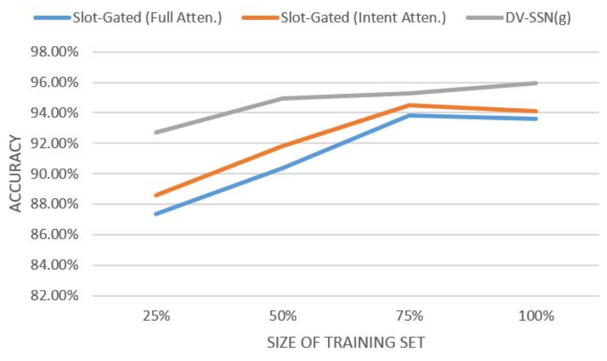

Fig. 7: The influence of training set size on performance (ATIS dataset)

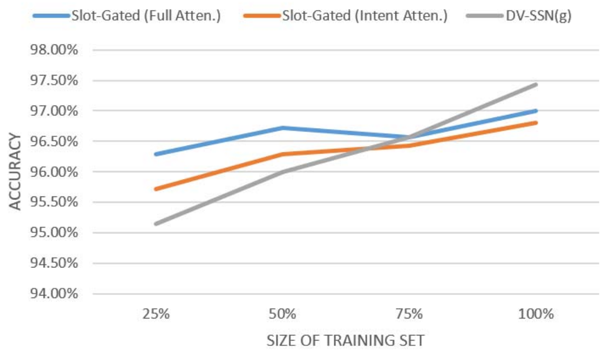

Fig. 8: The influence of training set size on performance (Snips dataset)

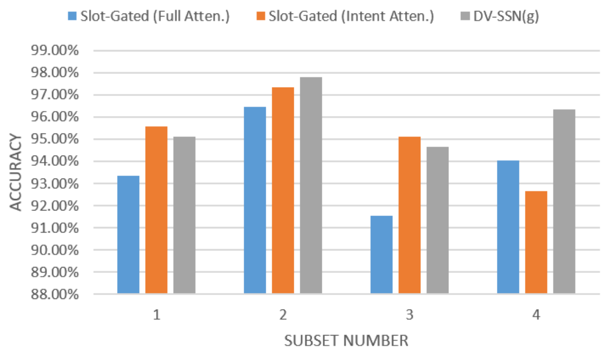

Fig. 9: Performance difference on different subsets (ATIS dataset)

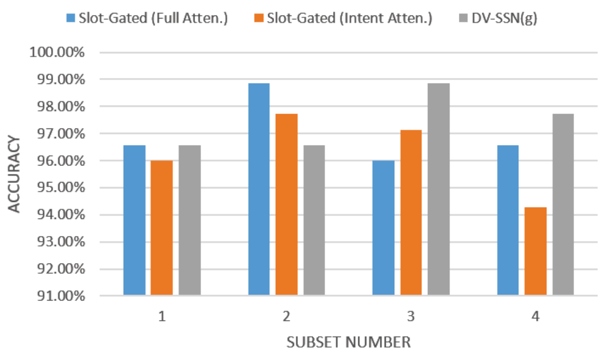

Fig. 10: Performance difference on different subsets (Snips dataset)

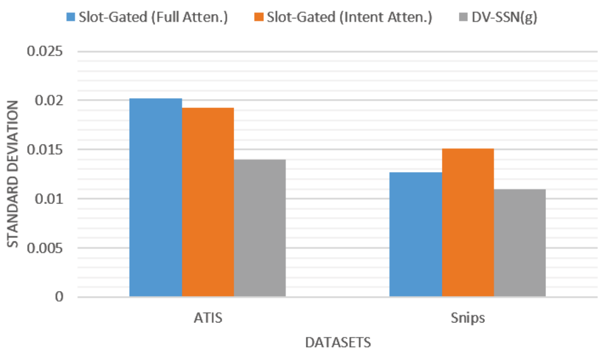

Fig. 11: The stability of the methods are measured by the standard deviation, and experiment results shows that our proposed method achieves the lowest standard deviation

V. Conclusion and future work

In this paper, we proposed a new method to do intent detection based on sentence embedding and support vector machine. Our system contains a frontend which is used to do sentence embedding and a backend which is used to do intent classification. We evaluated the performance of our method on two popular intent detection datasets. Experiment results show that our method outperforms previous methods in terms of accuracy and stability. What's more, our method also shows greater generalization ability on small size training set.

Although our method has shown its great ability in doing intent detection, it has not been able to do intent detection and slot filling jointly. Since the sentence embedding technique can also mark the importance of each word in a sentence, doing slot filling based on sentence embedding may be one of our future research directions.

References

[1] L. Cui, S. Huang, F. Wei, C. Tan, C. Duan, and M. Zhou, "Superagent: a customer service chatbot for e-commerce websites," *Proceedings of ACL 2017*, pp. 97–102, 2017.

[2] C. Tao, W. Wu, C. Xu, Y. Feng, D. Zhao, and R. Yan, "Improving matching models with contextualized word representations for multi-turn response selection in retrieval-based chatbots," *arXiv preprint arXiv:1808.07244*, 2018.

[3] P. Zhu, Z. Zhang, J. Li, Y. Huang, and H. Zhao, "Lingke: A fine-grained multi-turn chatbot for customer service," *arXiv preprint arXiv:1808.03430*, 2018.

[4] P. Khurana, P. Agarwal, G. Shroff, L. Vig, and A. Srinivasan, "Hybrid bilstm-siamese network for faq assistance," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 537–545, 2017.

[5] D. Hakkani-Tür, G. Tür, A. Celikyilmaz, Y.-N. Chen, J. Gao, L. Deng, and Y.-Y. Wang, "Multi-domain joint semantic frame parsing using bi-directional rnn-lstm.," in *Interspeech*, pp. 715–719, 2016.

[6] B. Liu and I. Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling," *arXiv preprint arXiv:1609.01454*, 2016.

[7] C.-W. Goo, G. Gao, Y.-K. Hsu, C.-L. Huo, T.-C. Chen, K.-W. Hsu, and Y.-N. Chen, "Slot-gated modeling for joint slot filling and intent prediction," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, vol. 2, pp. 753–757, 2018.

[8] C. Xia, C. Zhang, X. Yan, Y. Chang, and P. S. Yu, "Zero-shot user intent detection via capsule neural networks," *arXiv preprint arXiv:1809.00385*, 2018.

[9] J. Ramos *et al.*, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, vol. 242, pp. 133–142, 2003.

[10] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[11] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[12] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[13] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 2017.

[14] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, "Fasttext.zip: Compressing text classification models," *arXiv preprint arXiv:1612.03651*, 2016.

[15] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," *CoRR*, 2017.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[18] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, pp. 160–167, 2008.

[19] Y. Liu, C. Sun, L. Lin, and X. Wang, "Learning natural language inference using bidirectional lstm model and inner-attention," *arXiv preprint arXiv:1605.09090*, 2016.

[20] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding," *arXiv preprint arXiv:1703.03130*, 2017.

[21] H. Zhao, Z. Lu, and P. Poupart, "Self-adaptive hierarchical sentence model.," in *IJCAI*, pp. 4069–4076, 2015.

[22] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, 1992.

[23] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.