Conditional Joint Model for Spoken Dialogue System

Changliang Li[1(✉)], Yan Zhao[2], and Dong Yu[2]

[1] Kingsoft AI Lab, Beijing, China
lichangliang@kingsoft.com
[2] Beijing Language and Culture University, Beijing, China
zhaoyan.nlp@gmail.com, yudong@blcu.edu.cn

Abstract. Spoken Language Understanding (SLU) and Dialogue Management (DM) are two core components of a spoken dialogue system. Traditional methods model SLU and DM separately. Recently, joint learning has made much progress in dialogue system research via taking full advantage of all supervised signals. In this paper, we propose an extension of joint model to a conditional setting. Our model does not only share knowledge between intent and slot, but also efficiently make use of intent as a condition to predict system action. We conduct experiments on popular benchmark DSTC4, which includes rich dialogues derived from real world. The results show that our model gives excellent performance and outperforms other popular methods significantly, including independent learning methods and joint models. This paper gives a new way for spoken dialogue system research.

Keywords: Joint learning · Spoken language understanding · Dialogue management

1 Introduction

One long-term goal in artificial intelligence field is to build an intelligent human-machine dialogue system, which is capable of understanding human's language and giving smooth and correct responses. Especially with the success of new speech- based human-computer interfaces, there is a great need for effective dialogue agents, such as digital personal assistants, which can handle everyday tasks such as booking flights. SLU and DM are two essential parts in building a spoken dialogue system [1].

A typical dialogue system is designed to execute the following components: (i) automatic speech recognition converts a spoken query into transcription; (ii) spoken language understanding (SLU) component analyzes the transcription to extract semantic representations; (iii) dialogue manager (DM) interprets the semantic information and decides the best system action, according to which the system response is further generated either as a natural language output or a result page.

R. Xu et al. (Eds.): ICCC 2019, LNCS 11518, pp. 26–36, 2019.
https://doi.org/10.1007/978-3-030-23407-2_3

SLU aims to obtain semantic representations in user utterances. In SLU, there are two main tasks: slot filling and intent detection. Slot filling aims to assign a semantic concept to each word in an utterance. Intent detection aims to identify the intent that users express. DM is responsible for controlling the dialogue flow, tracking the dialogue states and deciding what actions the system should take to handle the interaction between users and system. For DM, we focus on system action prediction (SAP) in this work.

Traditional approaches train SLU model and SAP model separately, which may have restrictions on knowledge sharing. In order to take full advantage of all supervised signals and utilize the information from both tasks, some joint models have been explored [2,3]. However, the traditional way of joint learning is just combining the loss functions of slot filling and intent detection, which brings the limitation that the information is hard to be used and transmitted effectively. We consider that intent labels, slot tags and actions are correlated, and intent information is helpful for slot filling and SAP. We use intent information as condition to integrate with semantic representations for slot filling and SAP.

In this paper, we propose a conditional joint model that can be used to perform SLU and SAP. In our model, we obtain the semantic representations by a shared Bi-LSTM layer. Meanwhile, intent information is provided to predict slot tags, and is used as a condition to predict system action. Moreover, knowledge between the three supervised signals can be shared implicitly by joint learning. We evaluate our model on the popular benchmark DSTC4 dataset. The results show that our model has a great performance and outperforms other popular methods significantly.

The rest of our paper is structured as follows: Sect. 2 discusses related work, Sect. 3 gives a detailed description of our model, Sect. 4 presents experiments results and analysis, and Sect. 5 summarizes this work and the future direction.

2 Related Work

In this section, we introduce some previous work on SLU and SAP.

Firstly, SLU consists of two tasks: slot filling and intent detection. Traditionally, slot filling can be viewed as a sequence labeling task and intent detection can be viewed as an utterance classification task. These two tasks are usually processed by different models separately.

Machine learning methods such as hidden Markov models (HMM) [4] or conditional random fields (CRF) [5] have been widely employed for slot filling. These methods need complicated feature engineering. However, models with neural network architectures show advantages in feature generation and have a good performance on slot filling task. RNNs are applied in [6–8]. [9] utilized LSTM to generate context-aware distributions for capturing temporal dependencies. [10] enhanced the LSTM-based slot filling to model label dependencies.

Several classifiers such as SVM [11], Adaboost [12] and maximum entropy [13] have been employed for intent detection. With the development of deep learning, deep belief networks (DBNs) have been applied [14]. [15] proposed an RNN architecture to improve intent detection.

In recent years, joint learning of slot filling and intent detection are explored for utilizing some shared knowledge. [16] utilized CNN based triangular CRF to extract features for joint learning slot filling and intent detection. [17,18] adapted RNNS for joint learning of SF and ID. [19] presented a contextual method to exploit possible correlations among intent detection and slot filling. [20] utilized explicit alignment information in the attention-based encoder-decoder neural network models.

For SAP, [21] explored a partially observable Markov decision process (POMDP) to control the system actions. Furthermore, RNN based dialog state tracking models for monitoring the dialogue progress was proved [22]. [2] provided conjoint representations among the utterances, slot- value pairs and knowledge graph representations to overcome current obstacles of deploying dialogue systems. [23] implemented an online learning framework to jointly train actions and the reward model with a Gaussian process model. [24] employed a value iteration method of reinforcement learning framework. [25] described a novel framework using genetic algorithm to optimize system actions. [3] proposed an end-to-end deep recurrent neural network with limited contextual dialogue history to train SLU and SAP jointly.

3 Model

The structure of our conditional joint model is shown in Fig. 1. It consists of SLU model and SAP model. Firstly, SLU model takes user utterances as inputs and obtains the context-aware distribution of each word by a shared Bi-LSTM layer. Then it performs slot filling and intent detection through task-specific output layers. Using the hidden outputs from SLU model, a sentence-level distribution for each utterance is produced in SAP model, and the distribution is combined with intent information for predicting system actions.

3.1 SLU Model

As is shown in Fig. 2, SLU model consists of embedding layer, shared Bi-LSTM layer and task-specific output layers for slot filling and intent detection.

Embedding Layer. Given a sequence of words $w_1, w_2, ..., w_T$ as inputs, we map them into a vector space to produce embeddings $x = \{x_1, x_2, ..., x_T\}$, where x_t means the word embedding of the t-th word.

Shared Bi-LSTM Layer. We employ Bi-LSTM network to obtain the context-aware distribution of each word. Since we have x_t as the t-th word embedding,

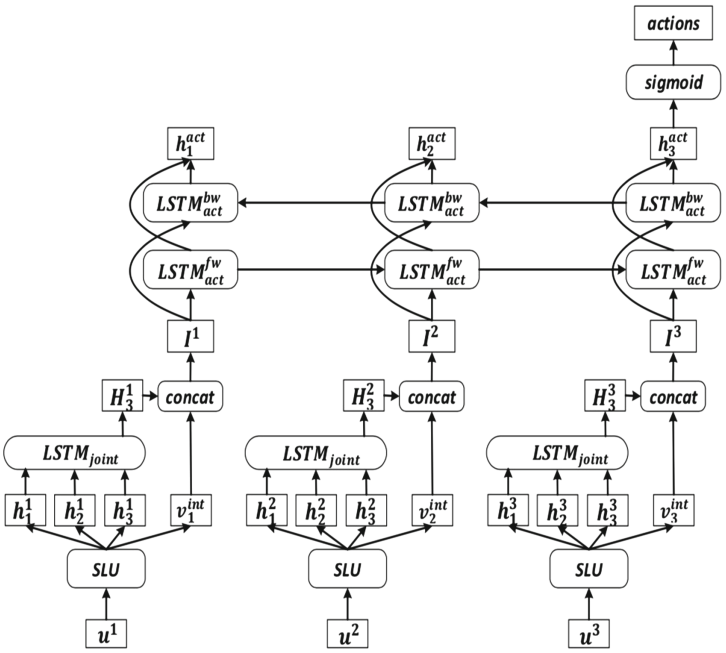

Fig. 1. Conditional joint model

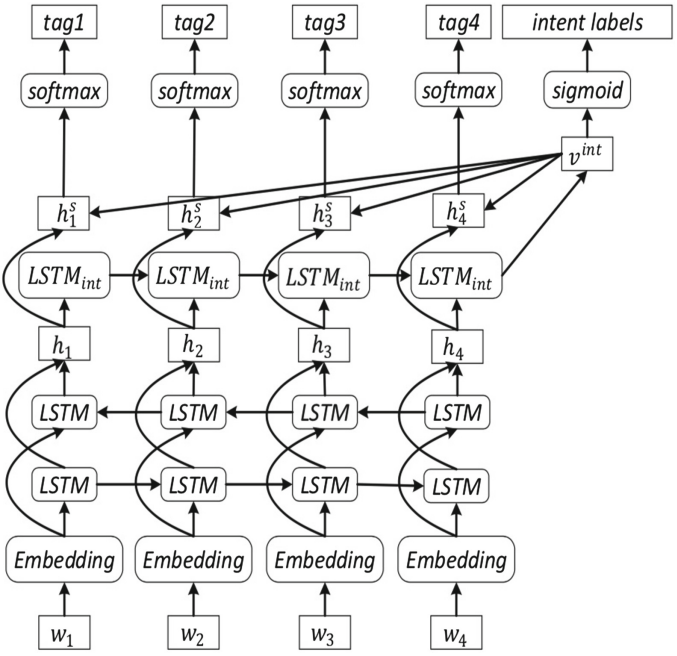

Fig. 2. SLU model

we calculate the forward and backward hidden states $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ respectively by the following equations,

$$\begin{aligned} i_t &= \sigma(W_{it}x_t + W_{hi}h_{t-1} + b_i) \\ f_t &= \sigma(W_{ft}x_t + W_{hf}h_{t-1} + b_f) \\ o_t &= \sigma(W_{ot}x_t + W_{ho}h_{t-1} + b_o) \\ \hat{c}_t &= tanh(W_{ct}x_t + W_{hc}h_{t-1} + b_c) \\ c_t &= f_t \odot c_{t-1} + i_t \odot \hat{c}_t \\ h &= o \odot tanh(c_t) \end{aligned} \tag{1}$$

where σ is the sigmoid function, $\odot$ is element-wise multiplication, and i, f, o and c represent input gate, output gate, forget gate and cell state respectively. W and b are trainable parameters. c_{t-1} means previous cell state; h_{t-1} means previous hidden state. Finally, we can obtain the final state h_t by concatenating the forward and backward hidden states. In this way, the context information is integrated from two directions as:

$$h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}] \tag{2}$$

Intent Detection Layer. We stack another LSTM layer $LSTM_{int}$ on top of the shared Bi-LSTM layer for intent detection,

$$h_t^{int} = LSTM_{int}(h_{t-1}^{int}, h_t) \tag{3}$$

where h_t is the hidden state at time step t. We take the last hidden state h_T^{int} for intent detection. Sometimes, there are more than one intents in a user utterance. In such a situation, we use a sigmoid function to calculate the probability over all intent labels,

$$p^{int} = sigmoid(W_T^{int} h_T^{int}) \tag{4}$$

where W_T^{int} is a weight matrix.

Similar with the intent detection layer, we obtain a threshold. The system action label is predicted if its probability is no less than the threshold,

$$y_n^{int} = \begin{cases} 1, & p_n^{int} \geq threshold \\ 0, & otherwise \end{cases} \tag{5}$$

$n \in [1, N]$ is the index of intent labels.

Conditional Slot Filling Layer. Since we have already obtained the last hidden state h_T^{int} from the layer $LSTM_{int}$, we use it as an intent vector v^{int}. The probability p_t^s is calculated as an attention weight to evaluate the contribution of the intent vector v^{int} to each hidden state h_t from the shared Bi-LSTM layer.

$$p_t^s = softmax(h_t \odot v^{int}) \tag{6}$$

Then, we add the hidden state and weighted intent vector together for predicting slot labels.

$$h_t^s = h_t + v^{int} * p_t^s \tag{7}$$

Finally, we choose the maximum of the probability as the predicted slot label,

$$y_t^s = argmax(softmax(W_t^s h_t^s + b^s)) \tag{8}$$

where W_t^s is a weight matrix.

3.2 Conditional Joint Model

To predict system actions, we joint SLU and SAP model together to make full use of information from each other. In multi-turn dialogues, history utterances play an important role in system actions. We recombine the utterances in a window size $u = \{u^1, u^2, ..., u^k\}$, where u^k is the k-th utterance in the window. Then we put them into SLU model for slot filling and intent detection. For each utterance, we can obtain the hidden outputs $h_t^k (t = 1, ...T, k = 1, ...K)$ and the intent vector v_k^{int} from SLU model, then we take h_t^k as inputs to a LSTM layer $LSTM_{joint}$ and use the last hidden state H_T^k to produce a sentence-level distribution.

$$H_t^k = LSTM_{joint}(H_{t-1}^k, h_t^k) \tag{9}$$

We concatenate the sentence-level distribution H_k with the intent vector v_k^{int} to utilize intent information.

$$l^k = [H_T^k, v_k^{int}] \tag{10}$$

Then the concatenated vector I^k is used as the input to the top Bi-LSTM layer for computing system action h_k^{act},

$$\begin{cases} \overrightarrow{h_k^{act}} = LSTM_{act}^{fw}(\overrightarrow{h_{k-1}^{act}}, I^k) \\ \overleftarrow{h_k^{act}} = LSTM_{act}^{fw}(\overleftarrow{h_{k-1}^{act}}, I^k) \\ h_k^{act} = [\overrightarrow{h_k^{act}}, \overleftarrow{h_k^{act}}] \end{cases} \tag{11}$$

where $LSTM_{act}^{fw}$ and $LSTM_{act}^{bw}$ stand for forward and backward LSTM network for SAP respectively.

The last hidden state h_K^{act} is used for predicting system actions. System can make more than one actions for an user utterance. Therefore, we use a sigmoid function to calculate the probability overall system action labels,

$$p^{act} = sigmoid(W_K^{act} h_K^{act}) \tag{12}$$

where W_K^{act} is a weight matrix.

Similar with the intent detection layer, we obtain a threshold. The system action label is predicted if its probability is no less than the threshold,

$$y_m^{act} = \begin{cases} 1, & p_m^{act} \geq threshod \\ 0, & otherwise \end{cases} \tag{13}$$

where $m \in [1, M]$ is the index of system action labels.

The loss function for SAP is defined as:

$$\mathcal{L}_{act} = -\sum_{m=1}^{M} a_m^{act} log y_m^{act} \tag{14}$$

where a_m^{act} means the ground truth label of system action.

In this joint model, the losses for slot filling and intent detection are defined as,

$$\begin{aligned} \mathcal{L}_{int} &= -\sum_{n=1}^{N} g_n^{int} log y_n^{int} \\ \mathcal{L}_{slot} &= -\sum_{t=1}^{T} s_t^s log y_t^s \end{aligned} \tag{15}$$

where N is the number of intent labels. For joint learning of SLU model and SAP model, we add the three losses together. The total loss is as follows.

$$\mathcal{L}_{total} = \sum_{D} (\mathcal{L} + \mathcal{L}_{int} + \mathcal{L}_{slot}) \tag{16}$$

where $\mathcal{D}$ means the number of sequences in the total dataset. Via joint learning with the united loss function, the shared hidden states can combine two tasks jointly. Furthermore, the correlations of the two tasks can be learned and promote each other.

4 Experiment

In this section, we conduct experiment on benchmark DSTC4 and give the experiment result and analysis.

4.1 Corpus

DSTC4 corpus contains several multi-turn dialogues collected from Skype calls between tour guides and tourists. It involves touristic information for Singapore in five aspects: accommodation, attraction, food, shopping, and transportation. In this paper, we use the DSTC4 corpus setting following [3]. The training set contains 5648 utterances, the validation set contains 1939 utterances, and test set contains 3178 utterances. The number of slot labels, intent labels and system action labels are 87, 68, 66 respectively. The statistic of DSTC4 is shown in Table 1.

Table 1. DSTC4 corpus setup in this work

Contents	Train	Dev	Test
Utterances	5648	1939	3178
Slot labels	87	79	75
Intent types	68	54	58

4.2 Training Details

For comparison purpose, we used the same training configurations as work [3]. The model is trained on all of the training data with its learning rate initialized to be 0.01. In order to enhance the model, we set the maximum norm for gradient clipping to 5 and dropout rate to 0.5

Table 2. Results for slot filling and intent detection

Model	Slot Filling(SF)				Intent Detection(ID)				SLU(SF+ID)
	F1	P	R	FrmAcc	F1	P	R	FrmAcc	FrmAcc
CRF+SVMs	40.50	61.41	30.21	77.31	49.75	52.56	47.24	37.19	33.13
BiLSTMs	46.15	54.63	39.96	76.84	47.48	52.19	43.55	39.96	36.38
JointModel	45.04	53.35	38.97	76.49	49.67	52.22	47.35	**42.20**	37.38
Con-Joint model	**49.32**	53.62	**45.65**	**78.41**	**49.81**	50.04	**49.59**	**42.20**	**38.01**

4.3 Metrics

Following the work [3], the performance of slot filling, intent detection and system action prediction are measured by token- level micro-average F1-score and frame-level accuracy(calculated only when the whole frame is correct).

4.4 Experiment Results and Analysis

We compare our model with the results from [3]. Table 2 shows results for slot filling and intent detection. There are previous works for SLU on DSTC4:

- CRF+SVMs: CRF for slot filling and LinearSVM for intent detection are training separately.
- BiLSTMs: A shared Bi-LSTM layer is provided for joint learning of slot filling and intent detection.
- JointModel: A SAP model stacks on top of a history of SLU models simply.

From Table 2, we can see that our model gains much increase in slot filling task. In term of F1 score, our model outperforms previous best result (BiLSTMs) by 3.17%. In term of frame-level accuracy, our model achieves 1.1% improvement compared with previous best result (CRF+SVMs). In intent detection task, our model also shows good performance. It outperforms previous best result (CRF+SVMs) by 0.06% in term of token-level F1 score, and achieves the same score in term of frame-level accuracy. For both slot filling and intent detection, our model outperforms previous best result (JointModel) by 0.63% in term of the frame-level accuracy.

From all the results above, we can conclude that with conditioned intent information, our joint model performs well in SLU. This can be explained that intent labels can provide more effective information for predicting slot tags. Table 3 gives the results for SAP. The models in the table are introduced as follows.

- SVMs: LinearSVM with features of one-hot vectors of aggregated slots and intents.
- BiLSTMs: A Bi-LSTM layer which takes the predicted slot label and intent label from NLU model as input for system action prediction.
- OraSAP (SVMs): LinearSVM with human annotated slot tags and user intents.
- OraSAP (biLSTM): A Bi-LSTM layer whose inputs are the same as Oracle-SAP.

Our conditional joint model outperforms all other models in token-level F1 score, especially in the recall value. Compared with the best result (SVMs), our model obtains 2.54% improvement in F1 score and 10.05% improvement in the recall value especially. Through combining intent information for SAP, the model can identify the most accurate action labels, which brings the recall value with obvious increase.

Table 3. Results for system action prediction

Models	F1	P	R	**FAcc**
SVMs	31.15	29.92	32.48	7.71
BiLSTMs	19.89	14.87	30.01	11.96
JointModel	19.04	18.53	19.57	**22.84**
OraSAP(SVMs)	30.61	**30.20**	31.04	7.65
OraSAP(biLSTM)	23.09	22.24	24.01	19.67
Con-Joint Model	33.69	**27.88**	**42.53**	18.25

We found that most user utterances in the dataset have more than one action labels (the maximum is 20). It is difficult to predict all the actions correctly. To purse high F1 score, we make a trade-off between token-level F1 score and

frame-level. We found that most user utterances in the dataset have more than one action labels (the maximum is 20). It is difficult to predict all the actions correctly. To purse high F1 score, we make a trade-off between token-level F1 score and frame-level accuracy. Therefore, it is reasonable that our model ranks a little lower in term of frame-level accuracy.

Above all, our conditional joint model has a great performance on both SLU and SAP. It can be interpreted that slot tags, intent labels and actions share knowledge with each other, and they promote each other via joint learning.

5 Conclusion

In this paper, we proposed a conditional joint model that can be used to perform spoken language understanding and dialogue management. Our model is capable of achieving knowledge sharing between slot tags, intents and system actions by utilizing intent information. Experiments on dataset DSTC4 demonstrate that our model has an excellent performance and outperforms other popular methods significantly. In future work, we intend to explore how to integrate information from the three different tasks explicitly for an enhanced joint model. Besides, we plan to extend our work to spoken language generation task for a more complete spoken dialogue system.

Acknowledgements. This research is supported by The National Key Research and Development Program of China (2016QY03D0501).

References

1. Chen, Y.-N., Celikyilmaz, A., Hakkani-Tur, D.: Deep learning for dialogue systems. In: ACL 2017, Tutorial, pp. 8–14 (2017)
2. Mrksic, N., OSeaghdha, D., Wen, T., Thomson, B., Young, S.: Neural belief tracker: data-driven dialogue state tracking. In: ACL 2017 (2017)
3. Yang, X., et al.: End-to-end joint learning of natural language understanding and dialogue manager. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5690–5694 (2017)
4. Wang, Y.-Y., Deng, L., Acero, A.: Spoken language understanding. IEEE Signal Process. Mag. **22**(5), 16–31 (2005)
5. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the International Conference on Machine Learning, ICML, pp. 282–289 (2001)
6. Yao, K., Zweig, G., Hwang, M.-Y., Shi, Y., Yu, D.: Recurrent neural networks for language understanding. In: Interspeech, pp. 2524–2528 (2013)
7. Mesnil, G., He, X., Deng, L., Bengio, Y.: Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In: Interspeech, pp. 3771–3775 (2013)
8. Yao, K., Peng, B., Zweig, G., Yu, D., Li, X., Gao, F.: Recurrent conditional random field for language understanding. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4077–4081 (2014)

9. Yao, K., Peng, B., Zhang, Y., Yu, D., Zweig, G., Shi, Y.: Spoken language understanding using long short-term memory neural networks. In: Spoken Language Technology Workshop (SLT), 2014 IEEE, pp. 189–194 (2014)
10. Kurata, G., Xiang, B., Zhou, B., Yu, M.: Leveraging sentence-level information with encoder LSTM for natural language understanding. In: EMNLP 2016, pp. 2077–2083 (2016)
11. Haffner, P., Tur, G., Wright, J.H.: Optimizing SVMs for complex call classification. In: 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP 2003), pp. I–I (2003)
12. Schapire, R.E., Singer, Y.: BoosTexter: a boosting-based system for text categorization. Mach. Learn. **39**, 135–168 (2000)
13. McCallum, A., Freitag, D., Pereira, F.CN.: Maximum entropy Markov models for information extraction and segmentation. In: ICML, vol. 17, pp. 591–598 (2000)
14. Sarikaya, R., Hinton, G.E., Ramabhadran, B.: Deep belief nets for natural language call-routing. In: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5680–5683 (2011)
15. Ravuri, S., Stolcke, A.: Recurrent neural network and LSTM models for lexical utterance classification. In: Sixteenth Annual Conference of the International Speech Communication Association, pp. 135–139 (2015)
16. Xu, P., Sarikaya, R.: Convolutional neural network based triangular CRF for joint intent detection and slot filling. In: 2013 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), pp. 78–83 (2013)
17. Guo, D., Tur, G., Yih, W., Zweig, G.: Joint semantic utterance classification and slot filling with recursive neural networks. In: IEEE Spoken Language Technology Workshop (SLT), pp. 554–559 (2014)
18. Zhang, X., Wang, H.: A joint model of intent determination and slot filling for spoken language understanding. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16) (2016)
19. Shi, Y., Yao, K., Chen, H., Pan, Y., Hwang, M.Y., Peng, B.: Contextual spoken language understanding using recurrent neural networks. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5271–5275 (2015)
20. Liu, B., Lane, I.: Attention-based recurrent neural network models for joint intent detection and slot filling. Interspeech **2016**, 685–689 (2016)
21. Young, S., Gasic, M., Thomson, B., Williams, J.D.: POMDP-based statistical spoken dialog systems: a review. Proc. IEEE **101**(5), 1160–1179 (2013)
22. Henderson, M., Thomson, B., Young, S.: Deep neural network approach for the dialog state tracking challenge. In: Proceedings of the SIGDIAL 2013 Conference, pp. 467–471 (2013)
23. Su, P.-H., et al.: On-line active reward learning for policy optimisation in spoken dialogue systems. In: ACL 2016, pp. 2431–2441 (2016)
24. Xu, Y., et al.: Policy optimization of dialogue management in spoken dialogue system for out-of-domain utterances. In: 2016 International Conference on Asian Language Processing (IALP) (2016)
25. Hang, R., Xu, W., Yan, Y.: Optimizing human-interpretable dialog management policy using genetic algorithm. In: 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU) (2015)