IMPLEMENTATION OF GENERIC POSITIVE-NEGATIVE TRACKER IN EXTENSIBLE DIALOG SYSTEM

Sangjun Koo, Seonghan Ryu, Gary Geunbae Lee

Pohang University of Science and Technology, Pohang, Republic of Korea
giantpanda@postech.ac.kr, ryush@postech.ac.kr, gblee@postech.ac.kr

ABSTRACT

Dialog state tracking is one of the most challenging tasks in the implementation of statistical Dialog Management (DM) systems. In development of a Korean dialog system, we implemented a generic tracking approach that can be used to extend a given initial set of system-output types. Our approach uses two methods: confidence estimation for error modeling, and dialog abstraction for dialog state tracking. We adopted a phoneme-sequence matching algorithm to estimate confidence for erroneous Korean user input. We also adopted a positive-negative model to abstract and generalize the effect of given user input and corresponding system output on dialog-state updating. Experiment result implies that our model can be used for dialog tracking without significant loss of performance. We implemented dialog system to verify that our approach is feasible in a practical Korean dialog system that can be adopted for other languages.

Index Terms— generic dialog tracker, positive-negative dialog tracker, extensible dialog system

1. INTRODUCTION

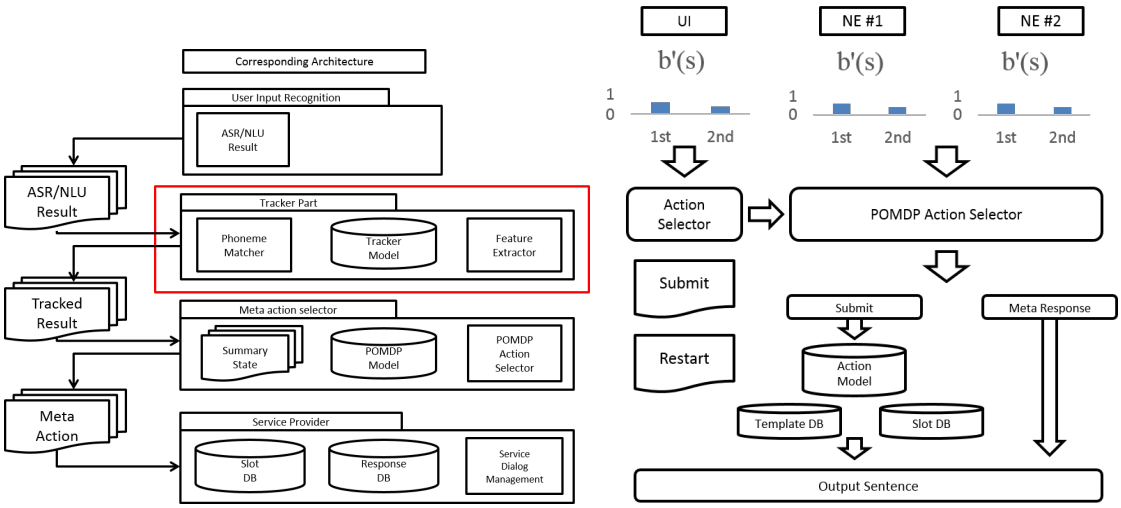
Partially Observable Markov Decision Processes (POMDP) have been adopted for statistical Dialog Management (DM) systems, because POMDP-based DM architecture allows designers to elicit mathematical models of user behavior, and thus the given architecture can manage input errors in an appropriate stochastic method [1][2]. Several practical DM systems have been implemented from various research projects because of this property [3][4].

One of the most challenging tasks in POMDP-based DM is to track and estimate the current dialog state of corresponding turns in a given dialog context. Automatic Speech Recognition (ASR) unit and Spoken Language Understanding (SLU) unit inevitably introduce errors, that cannot be distinguished from erroneous input. Hence, the POMDP architecture accepts a distribution of dialog states and generates optimal system output. For this purpose, an appropriate tracking algorithm with legitimacy should be developed, and it should have manageable computational complexity.

Several methods have been presented by studies in previous Dialog State Tracking Challenges [5][6][7]. Early studies proposed statistical architectures based on various techniques including Conditional Random Fields [8], Maximum Entropy Classfiers [9], and Neural Networks [10]. Rule-based deterministic approaches are also used to track dialog states. Given a sufficient amount of training data, statistical methods can reflect the diverse behavior of a given dialog system; this cannot be accomplished using deterministic methods. However, statistical methods are more computationally complex than rule-based methods, and thus the latter are preferable for implementation of practical dialog systems.

Often, researchers are restricted to working with insufficient amount of dialog log data and they may encounter several problems: (1) Given that they cannot retrain ASR/SLU directly without a sufficient amount of log data, how can they estimate the confidence scores of each entity from raw sequences when given ASR/SLU do not give them direct access to confidence scores? (2) Given that rule-based techniques cannot reflect diverse behavior without a massive amount of log data, how can we implement a dialog system that can be extended for new system-output types?

We present an architecture that can be used for generic tracking of belief states to solve presented problems. The architecture consists of two components: a confidence estimator and a positive-negative model ab-



(a) Architecture of hybrid POMDP DM (b) Architecture of meta-action selector (coupled)

Fig. 1: Schematic diagram of hybrid POMDP DM

stractor. The confidence estimator is used to estimate the probability distribution of given ASR/SLU input. We used Levenshtein distance to calculate the similarity between erroneous user input and corresponding slot entity. The positive-negative model abstractor is used to transform an arbitrary probability distribution, given by ASR/SLU, into positive-negative distribution, which allows the tracker to process various type of user input/system output pairs of a given turn.

2. HYBRID ARCHITECTURE

In this section, we explain our hybrid POMDP DM architecture (Fig. 1a) [11]. The architecture consists of three components: tracker, meta-action selector and service provider. When user input is interpreted as a probability distribution of slot entities, the tracker uses this distribution to track the belief states of each slot entity. The meta-action selector summarize the tracked belief state and uses the summary to guide selection of an appropriate meta-action including 'submit to service provider', 'confirm the value of slot x'. The selected meta-action is passed to the service provider, where a corresponding system action is generated. The dialog system is considered a 'hybrid' because it generates system actions not only from stochastic decision processes (meta-action selectors), but also from deterministic decision processes (service providers).

Since overall semantic information of sentences can be represented 'user-intention' and 'named-entity/slot-entity information', the meta-action selectors can be implemented with single POMDP models or by coupling user-intention POMDP models (UI-model) and slot-entity/named-entity POMDP models (NE-model). (Fig. 1b) Single POMDP models generate entire meta-responses from given overall belief states, whereas coupled models generate entire meta-responses from intermediate meta-response generated by UI-models and NE-models. Coupled architecture is more desirable for implementation because it requires a smaller number of belief states, which results in lower time-complexity in the training process. The method proposed (Fig. 2) in this paper is applied to the tracker component in the presented architecture (Fig. 1a).

3. METHODS

3.1. Mathematical Background

Given slot s, the aim of belief state tracking is to estimate belief state probability $b(s_t)$ for given turn t. Given observation o_t, and system output a_t, the belief state update rule for individual slot s can be written as [12]:

$$b(s_t) = \sum_{s_{t-1}, o_t} b(s_{t-1}) p(s_t | a_{t-1}, s_{t-1}, o_{t-1}) p(o_t) \quad (1)$$

The immediate goal is to solve two problems: process user inputs that have not been normalized by ASR/SLU,

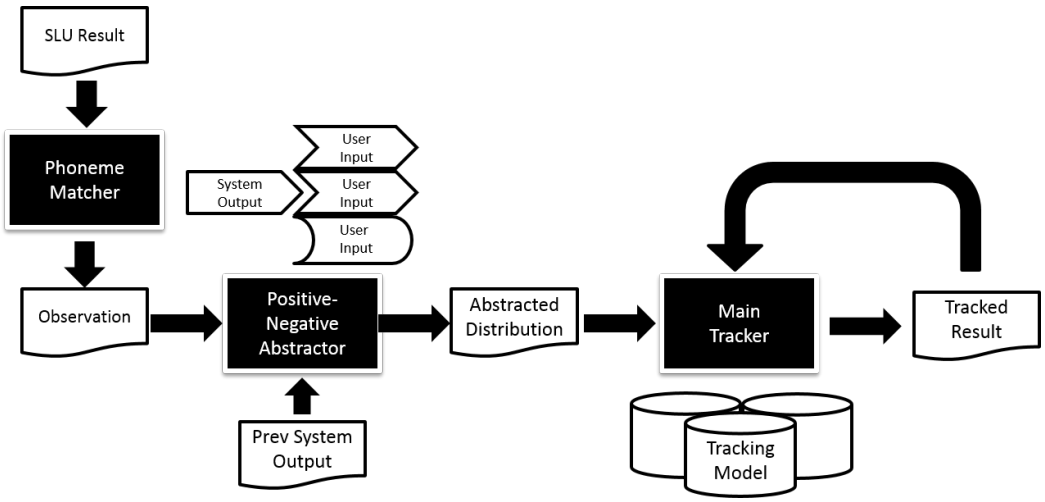


Fig. 2: Schematic diagram of proposed method

and manage various system output types from service providers. To solve the first problem, we assume an abstract component which that estimate observation probability from raw user input. To solve the second problem, we introduce positive factor s^+ and negative factor s^-. This strategy is inspired by previous studies [13][14].

Assume that o_t^{orig} represents the raw input sequences in turn t. By assuming independence between factors, the update rule can be re-written as below. Note that $p(o_t^{orig})$ represents a probability for input seqeunce itself and $p(o_t)$ represents a probability for slot-entity confidence.

$$b(s_t) = \sum_{s_{t-1},o_t} b(s_{t-1})p(s_t|a_{t-1},s_{t-1},o_{t-1})p(o_t) \quad (2)$$

$$= \sum_{s_{t-1},o_t^{orig}} b(s_{t-1})p(s_t|a_{t-1},s_{t-1},o_{t-1}) \quad (3)$$

$$\sum_{o_t^{orig}} p(o_t|o_t^{orig})p(o_t^{orig}) \quad (4)$$

$$= \sum_{s_{t-1},o_t^{orig}} b(s_{t-1})p(s_t|s_t^+,s_t^-,s_{t-1}) \quad (5)$$

$$\underbrace{p(s_t^+,s_t^-|a_{t-1},o_{t-1})}_{\text{positive-negative abstraction}} \underbrace{\sum_{o_t^{orig}} p(o_t|o_t^{orig})p(o_t^{orig})}_{\text{confidence estimation}} \quad (6)$$

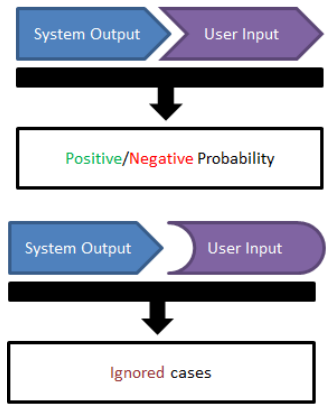
3.2. Confidence estimation

Converting ASR/SLU confidence to an observation distribution for specific slot entities is the most prior task to other operations in the system. Several techniques have been studied to implement improved ASR/SLU.

Error type	Error instance
No error(correct answer)	주군의 태양 Z UW G UW N WI TH EH JA NX
Confusion error	죽음의 태양 Z UW G WW M EY TH EH JA NX
Segmentation error (Over-segmented)	주군의 태양을 Z UW G UW N WI TH EH JA NX WW R
Segmentation error (Under-segmented)	주군의 태 Z UW G UW N WI TH EH

Table 1: Error types in processing Korean ASR/SLU

However, only some of them focus on the specific features of a featural alphabet. In Hangul, letters are clustered into syllables, but the assignment of sounds to syllables can be ambiguous, and thus several distinct error types can be observed in ASR/SLU (Table 1).First, there can be segmentation errors, when an ASR/SLU component fails to determine the boundary of given named entities at the syllable level which is not necessarily bound to word boundary unlike English or Chinese. Also confusion errors tend to occur at the syllable level, which is contrary to the case of English, where confusion errors tend to occur at the morpheme/word level. Because of this property, it is relatively more difficult to implement Hangul ASR/SLU units than to implement English ASR/SLU units.



System Output	User Input	Positive / Negative $p(s_{Jaws})$
Request-title	Inform (title=Jaws)	$p(s^+_{Jaws})$
Confirm (title=Jaws)	Affirm (title=Jaws)	$p(s^+_{Jaws})$
Confirm (title=Jaws)	Negate (title=Jaws)	$p(s^-_{Jaws})$
Recommend (title1=Jaws, Title2=Joe)	Select(first)	$p(s^+_{Jaws})$
Confirm (title=Jaws)	Select(first)	Ignored

Fig. 3: Triggering interpreting process with given rule-set defined for slot entity. (Title:Jaws)

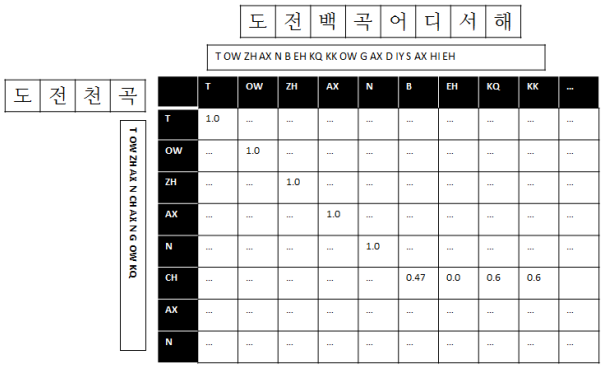


Fig. 4: Phoneme sequence similarity score calculation between user input "도전백곡어디서해" vs. slot entity "도전천곡".

We consider phoneme sequence similarity between a given user input and corresponding entities. Given vowel and consonant similarity between two phonemes, we can calculate a similarity score between two arbitrary phoneme sequences by measuring the Levenshtein distance between them. By normalizing the output with each slot entity (Fig. 4), we obtain an estimate of the confidence distribution.

3.3. Positive-negative abstraction

The motivation behind positive-negative abstraction is the generic description of belief state update rules, which is intended to construct linear solution for belief state tracking [13]. Given a user input and system output pair, (o_t, a_{t-1}), the result of the user response to the system output may increase or decrease the probability of a specific slot value. Stating the specific slot value ('the value of slot s is v.') and affirming the system confirmation ('Would the value of slot s be v'-'yes.') supports the related belief state, whereas denying the specific slot value ('the value of slot s is not v.') and contradicting the system confirmation ('Would the value of slot s be v'-'no.') opposes this state.

Positive-negative abstraction consists of two stages: Interpreting and belief state updating. Let s_v^+, s_v^- be random variables that indicates the degree of affirmation and negation, The interpreting process rewrites (o_t, a_{t-1}) into a form of s_v^+, s_v^-. For example, informationrule can be described as $P(inform(s = v)) \in P(s_v^+)$. The rule can be written as equations of specific user input and system output pair, (o_t, a_{t-1}). If the types in pairs fail to be matched, the corresponding rule would not be triggered. From this perspective, the *tractable* user action set consists of actions that can be *interpreted* by a specific rule. (Fig. 3)

Several techniques can be used for belief state tracking with a positive-negative model. Sun et al. suggested a linear solution that be described as a Markov Bayesian Polynomial model [13]. For simplicity, we use a linear formula (7) for heuristic tracking. Designers can also elicit statistical models with supervised training if they have the actual dialog log data set.

$$p(s_t|s_t^+, s_t^-, s_{t-1})p(s_t^+, s_t^-)b(s_{t-1}) \quad (7)$$

$$\approx 1-(1-b(s_{t-1}))(1-p(s_t^+))+b(s_{t-1})(1-p(s_t^-)) \quad (8)$$

Because the variables orignate from the marginal

probability distribution of user input and system output, the summmation of the discrete probability distribution over variables equals one, if we consider every possible user input-system output pair.

$$\sum_{v} p(s_v^+) + p(s_v^-) = 1 \tag{9}$$

The main advantage of this method is that it reduces computational overhead during the training phase. In our method, dialog system designers need only add the corresponding interpretation rule to adopt additional system output types, because the tracking model for the belief state is fixed. Although the proposed method would commit estimation errors, the update model can be calibrated to reduce the frequency of certain errors.

4. EXPERIMENT

4.1. Experiment Settings

We iteratively trained the POMDP model with a baseline tracker and observed the reward performance of each tracker-rule subset. The baseline tracker was implemented using heuristic rules [14]. Each training session started with a zero-initialized policy parameter vector. Each dialog batch consisted of five dialog instances. We increased the confidence error rate by 5% for every 100 batches for training UI model and 1000 batches for training NE model. Policy parameters were updated at the end of each batch. Each dialog instance consists of two, three, or four requests with corresponding constraints. Ten turns at most were permitted for each dialog instance.

The overall dialog was penalized for incorrect behaviors: requesting irrelevant slots, providing incorrect services, and confirming wrongly. Penalties differed for each case, but ranged from -5 to -50. The small amount of positive reward (1 - 5) was given if the system gave a correct responses, and zero was given otherwise. The reward total was initialized to zero for each dialog instance. Each reward value is supplied to the corresponding POMDP models: UI and NE. For training and evaluating, we used a simulator implemented by a request-constraint model [15], that was also used in our previous work [11].

The learning curves indicate that models converge to a certain level of reward value, which implies feasibility of our coupled architecture. (Fig. 5) (Fig. 6)

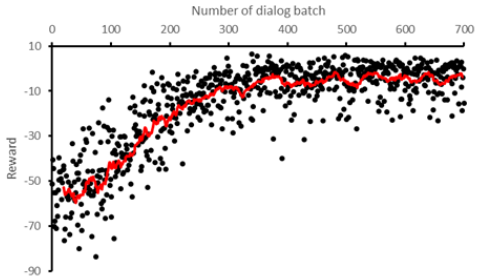


Fig. 5: Learning curve of UI model training. Red line: moving average for period 40

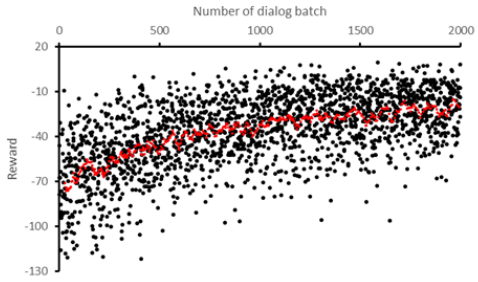


Fig. 6: Learning curve of NE model training. Red line: moving average for period 40

We measured average rewards for 200 batches with confidence error rate 35% for both pre-constructed UI and NE models. We tested three settings for the experiment: (1) using a baseline tracker, (2) using the proposed tracker with information rule only, (3) using the proposed tracker with information rule and confirmation rule(“confirm-affirm" and “confirm-negate").

4.2. Experiment Result

The results (Table 3) show the average reward value among three settings. The baseline tracker results show the highest average reward value for both the UI and NE models, -19.972 and -33.580, respectively. The results from setting 2 show the lowest average reward because the confirmation sentence and responses did not affect the overall tracking process. However, the difference between the baseline and given setting was not significant, because the simulator is intended to generate information responses eventually (e.g, “No, turn off the TV please."). The average reward values from setting 3 recorded almost equivalent levels, although there exists insignificant differences.

Turn	Sentence	Intention
System	무엇을 도와드릴까요? (How may I help you)	greet
User	TV 켜줘 (Turn on the TV.) TH IY B WW IY KH JX Z WX	request-tvOn
System	전원이 켜집니다. (It will be turned on.)	service-tvOn
User	TV **거죠** (Turn **o!@** the TV) TH IY B WW IY **K AX Z JO**	request-tvOff
System	수행하려는 기능이 TV 끄기가 맞습니까? (Did you intend to turn off the TV?)	confirm(UA=tvOff)
User	응 그래 (Yes, I did.) WW NX K WW R EH	affirm
System	TV가 꺼집니다. (It will be turned off.)	service-tvOff

Table 2: Sample dialog between user and proposed system

	Setting 1 Baseline	Setting 2 (information)	Setting 3 (information + confirmation)
UI Average Reward (35%)	-19.972	-20.690	-20.024
Difference (Baseline)	-	-0.796 (4.00%)	-0.1168 (0.59%)
NE Average Reward (35%)	-33.580	-35.950	-33.983
Difference (Baseline)	-	-2.37 (7.06%)	-0.403 (1.20%)

Table 3: Experiment results for confidence error 35%

5. IMPLEMENTATION

We implemented a dialog system that operates in the Electronic Program Guide (EPG) domain. The system is required to respond to various user requests for TV control, including "Find a program with a given title" and "Turn on the power." In the development of the EPG system, we used the Google speech recognition API for ASR and CRF sequence classifier for SLU. The recognized voice input from users was processed with a sentence normalizer and passed to the proposed system that uses the suggested architecture.

A sample dialog (Table 2) between a user and the proposed system shows the confirmation sequence of the user's intention. In the dialog, the user tries to turn off the TV. However, given user input is recognized incorrectly and is not interpreted to a proper sentence. Because of this error, the POMDP architecture opted asking the user for confirmation, which successfully results in the appropriate behavior (Turning off the power).

6. CONCLUSION

This paper presents a new method for tracking the belief state that can be adopted in practical dialog systems. We introduced positive-negative abstraction with confidence estimation in order to implement generic tracking components. The experiment implied that our proposed method can be compatible with a dialog system.

One further topic is model selection over belief state updating. Although we used a heuristic formula, the overall tracking performance can be improved by selecting the proper tracking model. From this perspective, training, selecting, and migrating the proper tracker model would be an interesting topic.

Acknowledgements

This work was partly supported by ICT R&D program of MSIP/IITP [R0101-15-0176, Development of Core Technology for Human-like Self-taught Learning based on a Symbolic Approach] and ATC (Advanced Technology Center) Program of MOTIE [10048448, Development of Conversational Q&A Search Framework Based On Linked Data]

7. REFERENCES

[1] Nicholas Roy, Joelle Pineau, and Sebastian Thrun, "Spoken dialogue management using probabilistic reasoning," in *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, Stroudsburg, PA, USA, 2000, ACL '00, pp. 93–100, Association for Computational Linguistics.

[2] Bo Zhang, Qingsheng Cai, Jianfeng Mao, and Baining Guo, "Planning and acting under uncertainty: A new model for spoken dialogue systems," in *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, San Francisco, CA, USA, 2001, UAI'01, pp. 572–579, Morgan Kaufmann Publishers Inc.

[3] Steve Young, Milica Gašić, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu, "The hidden information state model: A practical framework for pomdp-based spoken dialogue management," *Comput. Speech Lang.*, vol. 24, no. 2, pp. 150–174, Apr. 2010.

[4] Milica Gasic, Catherine Breslin, Matthew Henderson, Dongho Kim, Martin Szummer, Blaise Thomson, Pirros Tsiakoulis, and Steve Young, "Pomdp-based dialogue manager adaptation to extended domains," in *Proceedings of the SIGDIAL 2013 Conference*, Metz, France, August 2013, pp. 214–222, Association for Computational Linguistics.

[5] Deepak Ramachadran Jason Williams, Antoine Raux and Alan Black, "The dialog state tracking challenge," in *Proceedings of the SIGDIAL 2013 Conference*, Metz, France, August 2013, Association for Computational Linguistics.

[6] Matthew Henderson, Blaise Thomson, and Jason Williams, "The second dialog state tracking challenge," in *15th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2014, p. 263.

[7] Matthew Henderson, Blaise Thomson, and Jason Williams, "The third dialog state tracking challenge," in *Proceedings of IEEE Spoken Language Technology*, 2014.

[8] Sungjin Lee, "Structured discriminative model for dialog state tracking," in *Proceedings of the SIGDIAL 2013 Conference*, Metz, France, August 2013, pp. 442–451, Association for Computational Linguistics.

[9] Sungjin Lee and Maxine Eskenazi, "Recipe for building robust spoken dialog state trackers: Dialog state tracking challenge system description," in *Proceedings of the SIGDIAL 2013 Conference*, Metz, France, August 2013, pp. 414–422, Association for Computational Linguistics.

[10] Matthew Henderson, Blaise Thomson, and Steve Young, "Deep neural network approach for the dialog state tracking challenge," in *Proceedings of the SIGDIAL 2013 Conference*, Metz, France, August 2013, pp. 467–471, Association for Computational Linguistics.

[11] Sangjun Koo, Seonghan Ryu, Kyusong Lee, and Gary Geunbae Lee, "Scalable summary-state pomdp hybrid dialog system for multiple goal drifting requests and massive slot entity instances," in *2015 International Workshop Series on Spoken Dialogue Systems Technology*, 2015.

[12] Lukas Zilka, David Marek, Matej Korvas, and Filip Jurcicek, "Comparison of bayesian discriminative and generative models for dialogue state tracking," in *Proceedings of the SIGDIAL 2013 Conference*, Metz, France, August 2013, pp. 452–456, Association for Computational Linguistics.

[13] Kai Sun, Lu Chen, Su Zhu, and Kai Yu, "A generalized rule based tracker for dialogue state tracking," in *Proceedings of IEEE Spoken Language Technology Workshop 2014*, 2014.

[14] Zhuoran Wang and Oliver Lemon, "A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information," in *Proceedings of the SIGDIAL 2013 Conference*, Metz, France, August 2013, pp. 423–432, Association for Computational Linguistics.

[15] Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve Young, "Agenda-based

user simulation for bootstrapping a pomdp dialogue system," in *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, Stroudsburg, PA, USA, 2007, NAACL-Short '07, pp. 149–152, Association for Computational Linguistics.