Collaboration-based User Simulation for Goal-oriented Dialog Systems

Devin Didericksen[*] **Oleg Rokhlenko** **Kevin Small** **Li Zhou** **Jared Kramer**
University of Washington Amazon
`diderick@uw.edu` `{olegro,smakevin,lizhouml,jaredkra}@amazon.com`

Abstract

Building a goal-oriented conversational agent (CA) is arguably one of the most challenging, but potentially impactful, artificial intelligence problems. While reinforcement learning (RL) has emerged as a promising approach for the dialog policy optimization subtask, most research in this area is on the RL methods and assume that an existing user simulator can be used for training. Therefore, building a high quality user simulator remains a critical, but relatively understudied problem for *real-world* CA applications. Traditionally, user simulators are modeled at the individual user level, using a user's history to predict the next action (i.e., content-based). However, it is difficult to apply these approaches to a larger scale domains, where the action space is order of magnitude larger than most domains described in the literature. In this paper, we propose a collaborative-based user simulator that predicts the next user utterance based on what similar users said in similar situations. We apply our approach to the domain of a customer service and demonstrate its usability, even when using relatively simple similarity measures.

1 Introduction

There has been a recent increase in commercial applications of conversational agents (CA), primarily in the context of smart personal digital assistants and smart home controllers (e.g., Amazon Alexa, Apple Siri, Google Home, Microsoft Cortana, amongst others). Responding to this demand, much of this technology has been expanded to offer general CA services (e.g., Amazon Lex, Facebook Messenger Platform, Google Assistant, Microsoft Bot Framework, Watson Conversational Service). These services primarily provide APIs for specifying *frame-and-slot* semantics to track dialog progress and *finite state transducers* (FST) for dialog management, generally limiting applicability to relatively simple tasks requiring a small number of turns (e.g., setting a timer, selecting a song). Building technology capable of managing longer, mixed-initiative, increasingly complex dialogs (e.g., healthcare, customer service) has thus become an increasingly active research area [1].

Within the space of goal-oriented CA, reinforcement learning (RL) has emerged as a promising approach to dialog policy optimization [2]. Consider Figure 1, where we are modeling a multi-turn conversation between a *user* and *agent* as a Markov decision process (MDP). Specifically, at each time step t, the input user statement, $a_u(t)$, is first converted into a semantic representation known as the *user dialog act*, $u(t)$, via a natural language understanding (NLU) component. Combining this with the *user record*, $s_u(t)$, and dialog history, the current dialog state, $s(t)$, is estimated. Based on $s(t)$, the agent selects an action, $a(t)$, composed of a single dialog act, $a_d(t)$, that will determine the agent output utterance via the natural language generation (NLG) component and a set of situated actions, $\overline{a}_s(t)$, that interact with the external environment (e.g., order diagnostic test, issue refund). Within this formulation, RL can be used to learn the dialog policy $\pi(s(t)) \rightarrow a(t)$ assuming that a suitable reward function, $r(s, a, s') \rightarrow \mathbb{R}$, can be defined.

[*]this work was performed while in Amazon

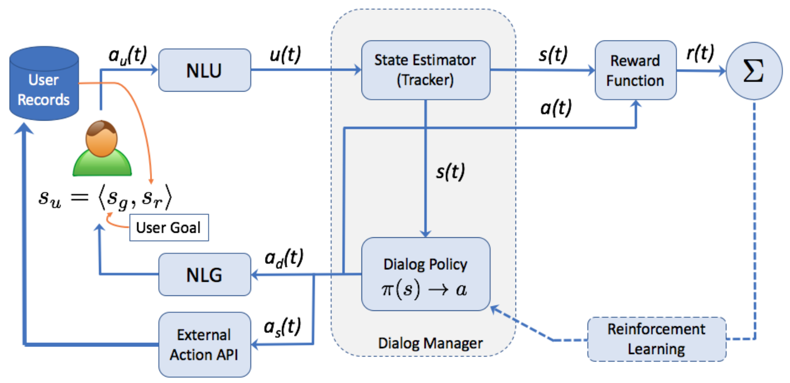

Figure 1: Markov Decision Process (MDP) formulation for learning a conversational agent dialog policy with reinforcement learning

This formulation results in the following technical problems with respect to building a dialog manager within a given domain: (1) deriving a suitable representation for the dialog state $s(t)$, (2) defining the dialog action space $a(t)$, (3) specifying a reward function $r(s, a, s')$ to learn the dialog policy, (4) developing a user simulator that can map $a_d(t) \rightarrow a_u(t)$, and (5) evaluating the resulting conversational agent. The MDP specification (i.e., (1)-(3)) is fairly straightforward to define in a goal-oriented slot filling system, making it the workhorse of most CA services, depsite recent research efforts to ameliorate the manual effort of defining slot-based states with RNN hidden state models [3]. Building a user simulator and evaluating the learned policy are less well-studied, but crucial to developing industry-scale RL solutions. This work specifically explores building a statistical user simulator from an existing corpus of dialog transcripts – specifically within a customer service setting.

Designing a high-quality user simulator is a difficult task with several academic research efforts (c.f., [4]). For use in practical RL-based setting, an effective user simulator should produce realistic, context-sensitive, and diverse responses to agent actions/utterances. Traditional user modeling methods tend to focus on modeling unobservable aspects of users including their beliefs, desires, and intentions (BDI) [5] and individual characteristics of users (e.g., expertise, cooperativeness, patience). Statistical models instead aim to construct a user simulator that generalizes over a broad space of user behaviors and responses [6]. However, ostensibly due to a dearth of *real-world* dialog transcripts, many of these works train on (semi-)synthetic data (e.g., bAbI tasks [3]), resulting in much of the state-of-the-art RL methods research depending on rule-based simulators. In this work, we take advantage of having a relatively large transcript corpus and develop a collaboration-based approach to user simulation – demonstrating promising results on a customer service application relative to a purely content-based method.

2 Customer Service Domain

The customer service (CS) domain is one of the application areas most amenable to automatic conversational agents due to the potential commercial and partially constrained action space. However, due to its relative complexity, limited work has been done in this domain. Specifically, the action space (both agent and user) is large compared to the more widely studied domains in the literature (e.g. restaurant reservation, flight booking, etc.) and the dialogs contain more mixed-initiative interactions. Recently, Williams et. al. [7] applied their RL-based approach to IT domain, which is similar to CS in some extent, but used a rule-based user simulator to train the agent – therefore significantly restricting their domain. To the best of our knowledge, there is no prior published art with respect to building a stochastic user simulator in the CS domain.

One distinguishing advantage of working on the CS domain is the large quantity of data available from existing human-to-human customer-agent conversations (at least within industry settings). Specifically, we used $> 100k$ dialog transcripts from a sub-domain of Amazon Customer Service spanning over a ≈ 2 year period, resulting in a total of $> 1.5M$ agent utterances and $> 1M$ user utterances. Analysis of the dataset revealed that even though the dialogs have mixed-initiative interactions, they are not symmetric from the linguistic point of view — the agents language is much more conservative and thus has a lower entropy. Presumably, this is a result of the CS agents participating in a training program intended to provide a relatively uniform customer experience with respect to mannerisms and business policies.

To validate this observation, we built a bi-gram language model for agents and customers and measured their *perplexity* (i.e., the average per-word log-probability on the holdout data set: $e^{-\frac{1}{N}\sum_i \ln p_{w_i}}$). We followed the standard procedure and summed over all the words of each utterance and then averaged over all utterances. Indeed, our analysis showed that while customers had a perplexity of 146, agents had a perplexity of 53. Computing median or so as changing n in the n-gram model didn't significantly change the overall results. Based on this observation, we focus on tracking dialog state via the agent actions, which are simpler to classify, to develop a collaborative-filtering approach for user simulation.

To better understand the agents actions space, we randomly sampled ≈ 1000 dialogs from our dataset and manually annotated the corresponding agent utterances. We used hierarchical annotation, where each agent act consisted of an *action* and a *specifier*. For example, the action $Salutation$ could have several possible specifiers — $Greeting$ for something like *"Hello, my name is <NAME>, I'm here to help you today."*; $Closing$ for something like *"Thank you for contacting Amazon and have a great day!"*; $Timeout$ for something like *"May I know are we still connected,please?"*, etc. This process resulted in hundreds of unique fully specified (i.e., action+specifier) agent actions, with a few dozens of higher-level actions (i.e., without specifiers).

3 Methods

Collaborative-filtering utilizes similarity between users to predict the user's next action. Early research in this area used a nearest neighbor approach to predict the user's next action. Most widely used by recommendation systems, the intuition can be stated as "the next user action has a high chance of being the same as similar users' action in the same state". Using this intuition for CA, we argue that if we would be able to identify similar users in the similar dialog state, we would predict next user utterance with a high confidence.

This observation reduces the user simulation problem to the user-to-user similarity problem, which is difficult to solve, especially in the dialog setting where we have to deal with sequences of utterances. The user action space is richer and every action can take very different forms (i.e. the language has a higher entropy). Instead, we propose to look at the agent actions space, and take advantage of the more conservative language with lower entropy.

In this section we suggest one, rather simple, approach that takes advantage of the assumptions above and show that even using such a simple method, we can obtain promising results. Our approach consists of two main stages: (Step 1) label propagation from the small annotated set to the entire corpus - see Figure 2(a); and (Step 2) finding the nearest neighbor in the corpus (i.e. a dialog that is most similar to the ongoing dialog and returning the next user utterance - see Figure 2(b)).

To propagate agent labels from the annotated set to the entire corpus, we trained a supervised multinomial classifier for predicting agent actions given an agent utterances. To this end we first cleaned the data using standard methods – tokenization, lemmatization, removing stop words, and substituting named entities with placeholders. We represented each utterance using BM25 on bi-grams and trained several supervised models - Multinomial Naive Bayes, SVM, and Random Forests. SVM with linear kernel provided best results with almost 83% accuracy, while two other models demonstrated slightly worse results. We also tried two RNN models — using LSTM units and GRU units with GloVe embeddings, but due to the small training set their performance was much worse.

After Step 1, each dialog is represented by a trajectory of agent actions. Given dialog $c \in C$, where each agent utterance $u_a(t)$ is annotated with a corresponding label $a_a(t) \in A$, $trajectory(c, i, \ldots j)$ is the sequence of all consecutive agent actions $< a_a(i), a_a(i+1), \ldots, a_a(j) >$. Also, bare in mind, that we still maintain the entire dialog structure, meaning that there is a user utterance $a_u(t)$ after each agent action $a_a(t)$, resulting in a sequence $< a_a(i) \rightarrow a_u(i) \rightarrow a_a(i+1) \rightarrow a_u(i+1) \rightarrow \ldots \rightarrow a_a(j) \rightarrow a_u(j) >$ – see Figure 2.

We use the procedure described in Algorithm 1 to select the next user utterance in the ongoing dialog. Specifically, $Similarity()$ is a function measuring the similarity between trajectories, $Rerank()$

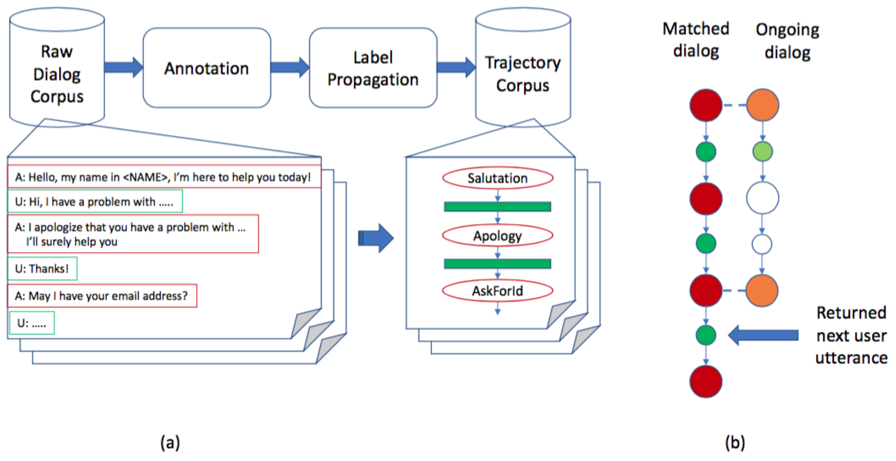
Raw Dialog Corpus
Annotation
Label Propagation
Trajectory Corpus
A: Hello, my name in <NAME>, I'm here to help you today!
U: Hi, I have a problem with .....
A: I apologize that you have a problem with ... I'll surely help you
U: Thanks!
A: May I have your email address?
U: .....
Salutation
Apology
AskForId
Matched dialog
Ongoing dialog
Returned next user utterance
(a)
(b)

Figure 2: Method overview. (a) Each raw dialog is converted into a trajectory representation by propagating agent labels from a small annotated corpus to the entire corpus. (b) Trajectory alignment between an ongoing dialog and a dialog from a corpus. Red/orange big circles represent agent actions, green/light green small circles represent user utterances. The alignment can include skips represented by empty circles. If this is the best matching dialog, the next user utterance is selected to be the return value of the user simulator for the next step.

is a function measuring the lexical similarity, and $GetUtterance()$ is a function returning the user utterance from the provided dialog.

Algorithm 1 Selects the next user utterance for the ongoing dialog

Precondition: d is an ongoing dialog, C is a corpus of dialogs

```
 1: $max\_similarity\_score \leftarrow 0$
 2: $P \leftarrow \emptyset$
 3: $tr_i \leftarrow trajectory(d, 1 \ldots |d|)$
 4: **for** $c \in C$ **do**
 5:     **for** $j \leftarrow 1$ to $length(c)$ **do**
 6:         $tr_j \leftarrow trajectory(c, 1 \ldots j)$
 7:         **if** $Similarity(tr_i, tr_j) > max\_similarity\_score$ **then**
 8:             $max\_similarity\_score \leftarrow Similarity(tr_i, tr_j)$
 9:             $clear\ P$
10:             $P \leftarrow (c, j)$
11:         **else if** $Similarity(tr_i, tr_j) = max\_similarity\_score$ **then**
12:             $P \leftarrow P \cup (c, j)$
13:         **end if**
14:     **end for**
15: **end for**
16: $Rerank(P)$
17: **return** $GetUtterance(P[0])$
```

In our settings, we used the inverse of Levinstein distance [8] for $Similarity()$; cosine similarity between the *tf-idf* representations of the last agent utterance in the ongoing dialog and the last agent utterance in the matched dialog for $Rerank()$, and we returned the user utterance following the last matched agent utterance in the matched dialog in $GetUtterance()$.

4 Results and Discussion

4.1 Results

In order to evaluate the performance of the methods above we established the following experimental design: the evaluator is presented with a partial dialog sampled at random from the entire corpus, truncated at a random agent utterance; then the evaluator is presented with two variants of the next user utterance — one generated by Algorithm 1 (while the selected dialog is excluded form corpus C) and another generated by the baseline. The baseline was a simpler variant of Algorithm 1 which uses only lexical similarity between the last agent utterances to find the nearest neighbor dialog. The order of both presented variants was also randomized. The evaluator should respond with either of the following options:

1. Both variants are plausible
2. Only variant A is plausible
3. Only variant B is plausible
4. None of the variants is plausible

After evaluating 200 dialogs, the baseline achieved a plausibility rate of 78.3%, while Algorithm 1 achieved a plausibility rate of 84.7%. The difference is significant with $p - value < 0.015$ using Wilcoxon signed-ranked test. The above results reveal several interesting findings: first, a simple information retrieval method to determine similarity between agent utterances already provide fairly good plausibility, meaning that many times the user response has Markovian properties (i.e., the user utterance is largely dependent on the previous agent utterance); second, trajectories alignment even in it's simplest form, like Levenshtein distance with the standard and equal weights, succeeds in capturing some structural properties of the dialog.

4.2 Discussion

Research on statistical user modeling usually distinguishes between *content-based* and *collaboration-based* form of modeling [9]. While most of the previously proposed approaches utilize content-based modeling, our approach has a collaborative-based nature. However, while traditional collaborative-filtering user modeling make use of a set of previous user experiences to find similar users, we use a sequence of utterances (or actions) in a particular dialog for this. We also argue that this sequence of utterances representation can be further abstracted by keeping only agent actions. There are several reasons for that: first, the agent action labels should be already available for a small set of annotated dialogs since it's a first step in MDP modeling of a dialog manager to define the agent action space; second, as we mentioned before, the agents' language has lower entropy and this is easier to model to accurately propagate labels.

Another common distinguisher between user simulation models is in the level of abstraction. There are 2 main types of models: word-level and intention-level. Even though our simulation model directly outputs the next user utterance, we argue that it still operates on the intention-level, rather than on the word-level. Since we are not trying to explicitly predict the next user action, we are using the intent of the user in the dialog which aligns the best with the ongoing dialog. Therefore, if we had the user intents explicitly annotated, we could operate on speech-act or dialog-act level and then generate the utterance using a NLG model. However, we choose to skip NLG and generate the utterance directly maintaining the intent implicit.

Even though our approach has some probabilistic elements, it's not a classical statistical model. Commonly, a stated advantage of probabilistic user models over other types of models is in their ability to create a "lifelike" randomness in user behavior. We argue, that our approach allows this randomness in a natural way by using a variety of real dialogs and applying randomization over the dialogs with the same degree of similarity. Additionally, it would not be difficult to draw from a distribution of nearest neighbors if diversity in responses is preferred.

Comparing the proposed approach to alternative methods for building user simulator, we would like to emphasize several points:

- When training an RL agent using the proposed approach, there is no need to use either NLU or NLG components

- The proposed approach can output auxiliary signals that can be used as a part of a reward function (e.g., a similarity score between trajectories or an indicator if the action chosen by the trained agent agrees with the real agent action in the matched dialog)
- The proposed approach is easy to implement and can be made to run very fast — an important aspect of RL training process.
- It's very easy to incorporate external knowledge into the proposed model (e.g., user profile can be used to pre-filter candidates for the trajectory alignment)

Finally, we argue that our approach is easily portable to other Customer Service domains. Once the agent action space is defined, it requires only a sufficient amount of annotated data for each label, which is more close to tens rather then hundreds examples due to the low perplexity of the agents language model. Also, many agent dialog acts are quite similar between the domain, especially the ones at the beginning and at the end of a dialog. Once the data is collected, the rest should be straightforward.

5 Related Work

With the increased interest in deep reinforcement learning [10], recently published research has largely focused on methodological advances over building scalable systems. However, there is a long history of user modeling within the computational linguistics and spoken dialog systems community [4]. Traditional linguistically motivated user simulators aim to construct a representative model of each individual user such that the CA can estimate the current state of the user and draws significantly from research on human-human interactions. By modeling user preferences, goals, knowledge, beliefs, capabilities, personality, sentiment, etc., the CA can consult this model to understand the intentions of the user and generate an appropriate response – and can also be used by the NLU and NLG components [9]. Operationally, these works are often dominated by knowledge-based formalisms [11] or rule-based systems [12]. From this perspective, the rule-based systems used in much of the recent RL for dialog research are fairly rudimentary.

Statistical models focus less on modeling individual users and instead emphasize building models that can generate user responses that are both realistic and context-sensitive. Often the goal of these methods is to provide feedback to a machine learned CA agent and thus prefer the utility of the response for the learned agent over a faithful representation of a human counterpart. In principle, these are preferable for scalable systems as they can be trained from existing human-computer dialog transcripts. By building a trainable system, it is easier to directly model context with respect to the current dialog state, generate realistic responses, and ensure sufficient variability in response to adequately explore the space of user utterances. Methodologically, there have been many statistical approaches to predicting the user response in a specified dialog state including Markov Models [13], Hidden Markov Models [14], general Bayesian Networks [15, 6], Decision Graphs [16], linear models [17], amongst others.

Instead of using user properties and their history of actions to predict future actions, collaboration-based models predict future actions based on actions taken by other users within the same context (i.e., dialog state). This changes the focus from collecting sufficient longitudinal data from individual users to collecting sufficient behavioral data (i.e., dialog transitions) for a population of users – thus making it amenable for large dialog corpora with a relatively constrained action space with a small number of dialog transcripts associated with each individual user (e.g., customer service at a company with high customer satisfaction). While not widely used for user simulation in CA settings [9], collaborative filtering has been widely used in the recommendation systems literature [18].

As previously indicated, recent work in CA research has focused on domains more highly constrained than customer service (e.g., restaurant reservations, travel reservations). However, there have been some notable efforts in building larger scale systems include IT customer support [7], online shopping [19] – highlighting some of the issues associated with scaling up solutions to more practical settings. Finally, while RL-based solutions have been prevalent in goal-oriented settings, there has been a general increase in neural machine translation-like architectures, both in task-oriented systems [3, 20, 21], open-domain systems [22, 23, 24], open-domain user simulators [25], and work that combines RL and neural models [26]. While we do internally integrate our user simulator with a deep RL method for learning a policy, this is beyond the scope of this paper.

6 Conclusions and Future Work

In this paper we proposed a simple collaboration-based approach for building a user simulator and implemented this within the customer service domain. Even though the CS domain is significantly richer than other domains usually used in the literature (such as restaurant reservations and flight booking), we demonstrated that based on certain observations on the data, it's possible to formulate the problem of predicting the next user utterance as a nearest neighbor collaborative filtering.

All-in-all we consider the plausibility rate of 84.7% as a good result for such a simple approach. We believe that it can be further improved by following a combination of the options below:

- Increasing label propagation accuracy: even though 83% accurate is reasonable, the performance of our classifiers suffered from training data imbalance — some of the labels were heavily underrepresented. Therefore, stratified sampling of relevant dialogs for annotation would increase the overall accuracy.
- Increasing trajectory alignment accuracy: in our settings we used a very simple form of sequence alignment, where every label had the same score for matching, deletion or insertion. Weighting these operations per label would increase the overall accuracy.
- Increasing contextual similarity accuracy: we use a cosine similarity between *tf-idf* representations of the last utterances to measure the contextual similarity. More complex models, like sentence embedding, e.g. [27], might improve the contextual similarity accuracy.

References

[1] 2017 NIPS workshop on conversational AI.

[2] S. Young, M. Gašić, B. Thomson, and J. D. Williams. POMDP-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, May 2013.

[3] Antoine Bordes, Y-Lan Boureau, and Jason Weston. Learning end-to-end goal-oriented dialog. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[4] Jost Schatzmann, Karl Weilhammer, Matt Stuttle, and Steve Young. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *The Knowledge Engineering Review*, 21(2):97–126, June 2006.

[5] Philip R. Cohen and Hector J. Levesque. *Rational Interaction as the Basis for Communication*, chapter 12, pages 221–256. MIT Press, 1990.

[6] Jost Schatzmann and Steve Young. The hidden agenda user simulation model. *IEEE Trans. on Audio, Speech, and Language Processing*, 17(4):733–747, 2009.

[7] Jason D Williams, Kavosh Asadi, and Geoffrey Zweig. Hybrid code networks: Practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 665–677. Association for Computational Linguistics, 2017.

[8] Robert A. Wagner and Michael J. Fischer. The string-to-string correction problem. *J. ACM*, 21(1):168–173, January 1974.

[9] Ingrid Zukerman and David W. Albrecht. Predictive statistical models for user modeling. *User Modeling and User-Adapted Interaction*, 11(1):5–18, Mar 2001.

[10] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[11] Robert C. Moore. Reasoning about knowledge and action. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 223–227, 1977.

[12] K. Komantani, S. Ueno, T. Kawahara, and H. G. Okuno. Flexible guidance generation using user model in spoken dialog systems. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2003.

[13] W. Eckert, E. Levin, and R. Pieraccini. User modelling for spoken dialgoue system evaluation. In *Proceedings of ASRU*, pages 80–87, 1997.

[14] Heriberto Cuayáhuitl, S. Renals, O. Lemon, and H. Shimodaira. Human-computer dialogue simulation using hidden markov models. In *IEEE Workshop on Automatic Speech Recognition and Understanding, 2005.*, pages 290–295, Nov 2005.

[15] O. Pietquin. *A Framework for Unsupervised Learning of Dialogue Strategies*. PhD thesis, Faculte Polytechnique de Mons, 2004.

[16] K. Scheffler. *Automatic Design of Spoken Dialogue Systems*. PhD thesis, Cambridge University, 2002.

[17] K. Georgila, J. Henderson, and O. Lemon. Learning user simulations for information statue update dialogue systems. In *Proc. of European Conference on Speech Communication and Technology*, 2005.

[18] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[19] Zhao Yan, Nan Duan, Peng Chen, Ming Zhou, Jianshe Zhou, and Zhoujun Li. Building task-oriented dialogue systems for online shopping. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 4618–4625, 2017.

[20] Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. A network-based end-to-end trainable task-oriented dialogue syste. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2017.

[21] Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. End-to-end task-completion neural dialogue systems. In *Proceedings of the International Joing Conference on Natural Language Processing (IJCNLP)*, 2017.

[22] Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. A neural network approach to context-sensitive generation of conversational responses. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 196–205, Denver, Colorado, May–June 2015. Association for Computational Linguistics.

[23] Iulian V. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3776–3783, 2016.

[24] Ganbin Zhou, Ping Luo, Rongyu Cao, Fen Lin, Bo Chen, and Qing He. Mechanism-aware neural machine for dialogue response generation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 3400–3406, 2017.

[25] Layla El Asri, Jing He, and Kaheer Suleman. A sequence-to-sequence model for user simulation in spoken dialogue systems. In *Interspeech*, 2016.

[26] Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. Deep reinforcement learning for dialogue generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1192–1202, 2016.

[27] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *EMNLP*, pages 681–691, September 2017.