Extracting Knowledge from Dialogue

Ondřej Plátek
Charles University in Prague, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Malostranské náměstí 25, 11800 Praha 1, Czech Republic
oplatek@ufal.mff.cuni.cz

Abstract

Building a conversation agent is a demanding process which is typically simplified by a using narrow and fixed conversation domain. The most effective approaches either use a very weak feedback and improve a deployed dialogue system via reinforcement learning or use supervised learning and labeled data. This work build on the success of the methods above and focuses on designing conversational agents which will be able to:

- collect explicit annotation interactively during the dialogue,
- enhance the knowledge base of a system by new facts,
- learn to recognize explicit reward signals in conversations.

Consequently, such conversational systems should:

- need smaller amount of data and annotation needed for their optimization,
- self-improve based on the collected feedback.

1 Introduction

The research of dialogue systems describes theories on how interlocutors communicate, communication techniques for humans and artificial systems are evaluated, and last but not how experimental artificial conversational systems are built. Arguably, the most understood and commercial successful artificial systems are conversation agents playing the role of an expert in task oriented dialogue in a narrow domain. Several research groups deployed such speech-to-speech dialogue systems on different domains, for example:

- Let's go system (Raux et al., 2005) helped the participants of experiments book a flight ticket and it was also deployed in Pittsburgh during night hours.
- The Cambridge group repeatedly uses Cambridge restaurant domain to evaluate experiments on crowd-sourced users where users search for a restaurant in Cambridge.
- The work of (Dušek and Jurčíček, 2016) and (Vejman and Jurčíček, 2015) evaluates their system on public transformation domain in Prague and New York, respectively, with crowd-sourced and also real users.

All the mentioned works conclude that action selection (the task of dialogue management) plays a central role in leading a dialogue. However, the obvious differences between domains and the absence of widely accepted evaluation metrics for action selection do not allow a comparison of the deployed techniques and algorithms.

The lack of comparable research was the reason for organizing the dialogue state tracking challenge (DSTC) (Williams et al., 2013), which resulted in successful evaluation of many dialogue state trackers on the restaurant domain. Dialogue state tracking represents the user's goal probabilistically within a predefined formalism such as dialogue acts (Williams et al., 2013, Henderson et al., 2014a,
Henderson et al., 2014b). A dialogue act is a triple $(type, slot'stype, slot'svalue)$ where type is *inform, confirm,...*, slot type is *food, area, ...* and values are for example *Chinese, west, ...* The dialog state tracker updates distribution over slots and their values as the conversation evolves. The DSTC evaluates the quality of the distribution by easy to understand and widely accepted measures such as accuracy and L2

measure.[1] The improvements of dialogue state trackers enable more informed and thus better action selection which is the ultimate goal of a dialogue system.

The dialogue state is commonly defined by a manually designed domain ontology. Thanks to the DSTC success, where a handcrafted ontology was provided, it may seem easy to design such an ontology[2]. However, we regard the manual onthology design as an arbitrary, costly and also error prone process. Recently, it was shown by (Wen et al., 2016) that dialogue state annotations are the only annotations in addition to conversation transcriptions needed for training an end-to-end system jointly, so it became even more important to specify high-quality DST labels by the domain ontology. In our recent work (Plátek and Jurčíček, 2016), we proposed alternative annotations of dialogue history, which we describe in Section 4.

We draw another conclusion from the dialogue state challenge: Using n-best lists from automatic speech recognition (ASR) helps just a little if compared with 1-best hypotheses even if ASR with high word error rates is in use. In addition, the recent advances in speech recognition reduced the WER even on far-field ASR drastically (Peddinti et al., 2015, Zhang et al., 2016). As a result, we observed that the focus of the research moved to text-to-text systems which can be easily integrated to speech to speech systems using a one-best ASR and a text-to-speech (TTS) modules.

We see the following research goals in the field of dialogue systems as the most important to address in next five years:

1. Reducing the number of data and annotation needed for deploying task-oriented dialogue systems.

2. Exploiting feedback and learned knowledge from live interaction with users.

3. Efficient exploiting knowledge gained from training a single domain agent for extending its domain.

[1]DSTC2 and DSTC3 challenges recommend using accuracy and L2 measures. In addition, one is advised not to evaluate the dialogue state trackers on the first turns where the dialogue state does not change.

[2]The Cambridge restaurant ontology had been polished over several years of research and it is actually rather simple.

We propose a research direction which aims to tackle the first two problems, and if successful, it may also help to solve the domain extension problem.

We review current state-of-the-art end-to-end dialogue systems in Section 2 with respect to our goals. In Section 3, we summarize how feedback is used for optimizing statistical models and how it is represented and extracted. In Section 4 we describe our so far results, and we propose a plan of our future research in Section 5. Finally in Section 6, we review our approach, discuss potential challenges and compare it to current research.

2 End-to-End Conversational Agents

Building end-to-end conversation agents is a new appealing approach for building conversation agents. It reduces the task of building a text-to-text dialogue system to training a single statistical model and thus optimizes the response generation process jointly with language understanding and state tracking and avoids accumulating errors along the pipeline.

Neural networks dominate the first attempts (Williams and Zweig, 2016, Bordes and Weston, 2016, Dodge et al., 2015) to build end-to-end conversational systems. Using neural networks is an obvious choice because various neural network models achieve state-of-the-art results for optimizing traditional components of a dialogue system:

- language understanding (LU) (Mairesse et al., 2009)
- dialogue state tracking (DST) (Williams, 2014, Henderson et al., 2014c, Vodolán et al., 2015, Plátek et al., 2016)
- natural language generation (NLG) (Dušek and Jurčíček, 2016, Wen et al., 2016)
- feedback/reward prediction (Su et al., 2015)

Neural networks models not only achieve the best results in all the mentioned tasks but the models are relatively straightforward to combine and optimize jointly. The key to successfully training a neural network is to provide enough labeled data since the neural network typically learns a lot of unspecified structure and patterns. Luckily, neural networks capture structures of data easily so less features need to be handcrafted and more importantly a sequence of components can be replaced

by a single statistical model. As consequence, the data for intermediate representation between the components is not needed for training the pipeline, but typically a larger amount of training examples is required.

2.1 Data annotation and loss function

Neural networks are commonly trained using supervised learning by maximizing directly the log likelihood of a conditional distribution $P(Y|X)$ where the parameters of the distribution are estimated from training samples $(\hat{X}, \hat{Y})$. Sequential problems such as dialogue conversation are formulated as a problem of generating the next reply y_t given the history h_t. In the case of a dialogue system, the history is represented by a sequence of the previous system utterances and user responses $y_1, x_1, y_2, \dot{x}_{t-1}, y_{t-1}, x_t$.

Similarly to Hidden Markov Models (HMMs) (Huang et al., 1990), Recurrent Neural Networks (RNNs) deal with potentially unlimited history by introducing a latent state s_t.[3] A simple use of RNNs (Gers et al., 2000) for classification resembles the structure of HMMs for maximum likelihood estimation, e.g. , in ASR (Huang et al., 1990), but RNNs model the observation probabilities discriminatively (see Figure 1). The probability of action y_t is computed based on parameters for updating previous state s_{t-1} to new state s_t based on observation x_t.

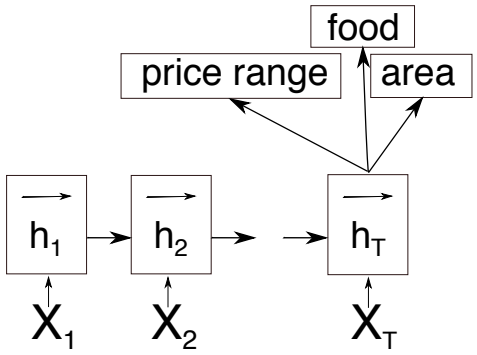

Figure 1: RNN encoder for classification. The output variables are conditioned on the hidden state. This model was used as a baseline in (Plátek et al., 2016) and it classifies DST state slots based on the last hidden state h_t independently.

In the work of (Wen et al., 2016), a neural end-to-end network model is trained for predicting a next system reply from a dialogue history. The model is trained not only from dialogue conversation transcriptions and the domain DB, but it also uses dialogue state annotations. The annotations are used because the model trains a pipeline of two neural networks and the annotations are used as the data representation of the first network output and the input of the second neural network. Its dialogue history is expressed as $h_t = y_1, x_1, s_1, y_2, \dot{x}_{t-1}, y_{t-1}, s_{t-1}, x_t, s_t$ where s_t is the annotation of dialogue state slots and their values e.g. *(food=Chinese, price_range=expensive, area=west)*. The work optimizes the model using supervised learning by maximizing the likelihood of the next word in a reply using cross-entropy training.

Cross-entropy training is the most successful method of training neural networks; However, maximizing the likelihood of the next word in a response should not be the ultimate goal which conversational agents should achieve. First, supervised approaches are limited by the quality of the golden data. Second, the independence assumptions of generating the next word from the hidden state are not true in dialogue. Consequently, there is plenty of room for improvement in modeling dialogue by end-to-end models.

Maximum likelihood: the most popular approach

Training neural networks for dialogue from gold labels using cross entropy for supervised learning brings several challenges:

- predictions of the next decision using the sequence-to-sequence model (Bahdanau et al., 2014, Sutskever et al., 2014) are conditionally independent given the sufficient statistics of the hidden state of the decoding RNN.

- Dialogue responses of the system are often ambiguous and multiple options are equally possible. On the other hand, cross-entropy loss function models uncertainty and ambiguity in data in the same way, allowing only one correct option. In Equation 1, $p(x)$ is a distribution approximated by the distribution of samples over training data and updates of the neural networks need to be performed only as difference between the predicted likelihood

[3] Unlike HMMs, they do not track probabilities in the hidden states but only use sufficient statistics.

$-\log(q(x))$ and a log one-hot distribution[4]:

$$H(p,q) = \sum_x p(x) * (-\log q(x)) \quad (1)$$

- Annotations of training pairs are costly and much more data is needed when annotations e.g. of dialogue state are not provided.

Weak supervision for reinforcement learning

In this section, we introduce reinforcement learning (RL) as a tool of our choice for optimizing the evaluation/reward/loss function e.g. user satisfaction or mimicking the whole dialogue. Reinforcement learning is a general framework for updating statistical model parameters even when a feedback for system's action is delayed or noisy (Williams and Zweig, 2016, Bahdanau et al., 2016, Wierstra et al., 2010). In the field of dialogue systems, the RL was successfully used to optimize parameters of Gaussian processes for selecting among several dozens of actions (Gasic et al., 2011). On the other hand, using relatively noisy feedback from a few thousands of live conversations with user feedback at the end of dialogues is not convenient for training neural networks because they need to update a large number of parameters and such feedback is too weak.

Note that RL was successfully used with supervised cross-entropy pre-training and later optimizing for not differentiable loss function. (Williams and Zweig, 2016). However, such approach still uses the same labeled data and annotations as used for supervised learning. The advantage of this approach is that it is able to naturally capture via the reward functions multiple valid actions and optimize directly the evaluation function. Furthermore, multiple weak reward functions can be combined to form much stronger feedback signal (Abbeel and Ng, 2004).

Reinforcement learning performance quality depends on the reward information which can be used for updating parameters of the model and also on the amount of data available. Typically the reward is also used as a scoring function for evaluating dialogues. However, if using the scoring function as reward function for reinforcement one would like to automate the computation, but only partial and not satisfactory automatic scoring functions were suggested (Liu et al., 2016, Lowe et al., 2016) for dialogue systems. Note also that some RL algorithms such as Sarsa are on-policy algorithms and need to be trained using live deployed systems. We focus only on algorithms which can be used in off-policy settings (Sutton and Barto, 1998) and does not require deployed system for their training.

Learning to collect feedback

The use of reinforcement or supervised learning assumes that the loss function is provided before the training. One of the key properties of dialogues systems is that their domains change over time (Yu et al., 2016 04), so every predefined loss function becomes obsolete after some time. In addition, specifying a good loss is notoriously hard since no standard measures for dialogue are widely accepted.

Inverse reinforcement learning (IRL)[5] is a task of learning the loss (or reward) function used in RL (Abbeel and Ng, 2004). IRL could be used for learning how to interpret user feedback, but the IRL is ill-posed problem in general (Choi and Kim, 2011) and is guaranteed to work only for special cases e.g. (Abbeel and Ng, 2004, Choi and Kim, 2011). In our future work (see Section 5) we aim to learn feedback from an interactive conversation, but we plan to use IRL and similar approaches as reward shaping (Su et al., 2016) only for comparison. We describe alternative approaches of collecting explicit annotation in Section 3.

Note, that collecting feedback explicitly for later use and IRL is not mutually exclusive. A promising framework which may overcome vagueness of current system is adversarial networks, especially work of (Dumoulin et al., 2016)[6], which remotely resembles inverse reinforcement learning. The adversarially learned inference (ALI) model is a deep directed generative model which jointly learns a generation network and an inference network using an adversarial process. The core idea is that one trains a generative model for dialogue system operator together with discriminator which attempts to recognize whether a generated dialogue is from the trained system or data sample from human (see Figure 2).

We are also interested in this approach because

[4] One-hot distribution is a categorical distribution where the single option is marked as gold and is assigned probability of one.

[5] Also known as active reward learning (Su et al., 2016).

[6] Picture is taken from `http://ishmaelbelghazi.github.io/ALI/`.

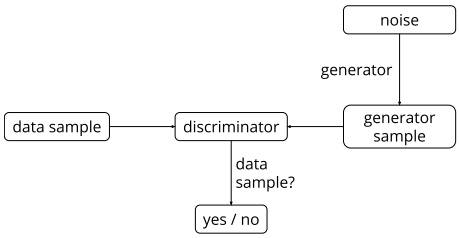

Figure 2: Adversarial Learned Inference core idea (Dumoulin et al., 2016) is to train a discriminator and a generator as one network with two objectives. Discriminator separates real samples and generated samples and generator produce such artificial samples so that the discriminator cannot distinguish them from the real one. The algorithm was demonstrated on CIFAR image dataset and needs to be adapted for generating dialogues (Krizhevsky et al., 2014).

one should be able to see which positive examples contributed the most for the performance of the generator (the dialogue system) by exploring the error updates and performance of the discriminator. Thus the discriminator could provide us with automatic annotations.

2.2 Architectures for neural end-to-end dialogue systems

Neural architectures are used in two use-cases of conversational agents;

- Chatbots — Such systems are trained without any structured knowledge from plain conversations.
- Task oriented systems — The systems learn to play a role of an expert and provide information from a database to users. They are trained in respect to the specific content of their database.

The work of (Serban et al., 2016) is an example of neural network architecture for chatbots which is able to mimic human-to-human conversations however quite often fails at capturing its semantics. This line of research improves upon language modeling using neural networks (Mikolov et al., 2013) and adapts modeling of next word prediction by more efficiently representing the discourse structures. The training of such models requires large corpora such as (Lowe et al., 2015).

On the other hand, researches working with task-oriented dialogue system train the models to extract common knowledge and discourse structures in the dialogue with help of a database containing domain related information. A limited domain also implicates that one is able to use only a limited amount of data for training the systems. The true challenge is how to model access to the system calls to the database because the calls are not recorded in the data since only plain text transcriptions of the dialogue are easy to acquire. One can only deduce that a part of the response is a result from some database call.

The work of (Wen et al., 2016) solved the problem by using the annotation of dialogue state which determines database calls and corresponding natural language response convenient for the dialogue state. The simplistic system (Williams and Zweig, 2016) showed that it is possible to use API calls[7] without dialogue state annotation, but the system required *action mask* heuristics which determined whether an action is possible for the dialogue context.

3 Collecting Feedback

Extracting feedback from dialogues is possible in several ways as described earlier: through fixed loss function, through learned loss function or explicitly through annotations. The feedback can be stored either explicitly as annotations or in parameters of the model. We focus on collecting explicit feedback because it can be exploited by any model via supervised or reinforcement learning. Nowadays, the interactive feedback from users is either ignored or stored only in parameters of statistical models e.g. neural networks. The crucial problem is that information represented in statistical models such as neural networks is notoriously hard to reuse for new architectures (Oquab et al., 2014). To our knowledge, we will be the first trying to extract the explicit feedback automatically. We consider it important task for automating dialogue systems because non-trivial expert work is needed for designing the heuristics or laborious effort is put into collecting annotations.

4 Experiments

Our work has focused on building an end-to-end conversation agent which eliminates labori-

[7]The system performs also other actions than accessed database so the term API call is used instead of DB calls.

ous handcrafting but needs a large portion data for training.

We first describe our published works, then we introduce work in progress and finally we propose future work in the next Section 5.

4.1 Published work

Our work so far has focused on developing an end-to-end task-oriented conversation agent on restaurant domain which is easy to train and provides a reasonable baseline. First, we verified that we are able to train recurrent neural networks for dialogue state tracking and achieve near state-of-the-art easily (Plátek et al., 2016). We frame the DST as sequence-to-sequence problem and used encoder-decoder (see Figure 3).

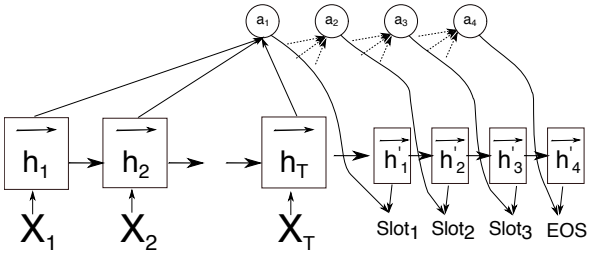

Figure 3: DST using an encoder-decoder RNN model.

Second, a dataset for training end-to-end system (Plátek and Jurčíček, 2016) was collected which focus on collecting easy-to-obtained annotations. The dataset contains both human-to-human conversations and corresponding calls to database. The crowdsource workers, which play the role of the system, annotate their response additionally which row of database they have used for answering the user query (see Figure 8). We argue that DB calls are much more natural and easy to obtain annotations both from crowd-sourcing and from life-system.

4.2 Work in progress

We are currently working on an end-to-end task-oriented system which can be optimized as single component. In contrast with very recent papers, our approach will hopefully need less annotations and doesn't use dialogue state labels as in (Wen et al., 2016), but it will use easier to obtain DB calls records (Plátek and Jurčíček, 2016).

A crucial problem of end-to-end statistical models for a dialogue system is that the dialogue agent performs latent actions which appear rather stochastically due to lack of training data, but the latent actions follow strict logic. The challenge for end-to-end models is to learn such logic in purely data-driven approach. An example of such (partially) latent actions are calls to database.[8] A system quering a database to obtain some information is a partially latent action because the system presents only the results of the action to the user. However, the user without the access to system database is not able to distinguish if the system is providing a valid answer. The system for example may choose credible reply based based on its language model (see Figure 5) which is plainly false. An example of completely latent action is if the user "*orders pizza*". The system should insert the order into the reservation system represented in DB. However, the user can only hope that the system executed the correct action and has to continue in the conversation. In the "*pizza order*" example, he or she happily waits at least 30 minutes for pizza delivery and if the dialogue system misunderstood the user and did not order the pizza, the user won't provide any feedback from which the system is able to learn. As a result, if one wants to train a dialogue system one needs to design objective function and data representation which force the latent action correspond to system replies, so the logic of the latent actions is maintained. Unfortunately, the feedback that a DB call does not correspond to the system reply is often not available as demonstrated in our "*pizza*" example.

Typically, the dialogue manager (DM) updates the dialogue state, presents the results of DB call and suggests an NLG plan - all represented in a discrete format such as dialogue state items (DAIs) (see Figure 4). The DM policy executes operations on KB and produces an NLG plan which is compatible with KB actions (Dušek and Jurčíček, 2016, Young et al., 2010). An alternative simplistic approach is presented in (Wen et al., 2016) where a KB operation is executed or skipped based on the dialogue state also represented by DAIs but the NLG uses only the dialogue state and the result of KB operation to generate a reply where the KB calls are reduced to simple select statement with single variable. We propose to predict directly which KB operation to use from the hidden state

[8] Any action of an agent can be transformed to a database call because the dialogue system may delegate the task through the database to other services which execute the actions. Typical example is a weather information service.

Model	Dev set	Test set
EncDec	0.867	0.730
(Vodolán et al., 2015)	-	0.745
(Žilka and Jurčíček, 2015 07 13)	0.69	0.72
(Henderson et al., 2013)	-	0.737
DSTC2 stacking ensemble (Henderson et al., 2014a)	-	0.789

Table 1: The Accuracy of our DST encoder-decoder compared to other implementation. The first group contains our systems which use ASR output as input, the second group lists systems using also ASR hypothesis as input. The third group shows the results for ensemble model using ASR output and also live language understanding annotations.

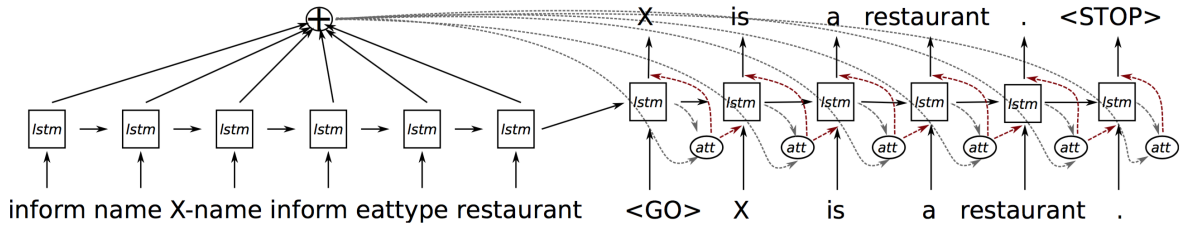

Figure 3: Seq2seq generator with attention

Figure 4: Dialogue act items tracking and generation. A policy ensures that each access to DB corresponds to NLG plan what to say (Dušek and Jurčíček, 2016).

h_t of neural network which uses distributed representation and later predict the system reply. The advantage is that one does not need to handcraft additional layer for tracking the dialogue state. In our approach, the predicted reply is conditioned on the selected KB call's result and the same state h_t state and will be generated in word by word manner. Such model needs to be trained so the replies match the KB calls, but it allows flexibility how to communicate about the KB calls and their results. We conditioned the reply based on the KB call and not vice versa, because we want the system to describe which action is actually used.

Experiments with our first model showed that we have too few data and our model is too complicated to learn anything meaningful on the DSTC2 dataset. In the experiment, we used the DSTC2 dataset and we tried to mimic the system replies given the dialogue history and the DB of restaurants for the DSTC2 restaurant domain. We design a heuristic function to create annotations in form of DB calls to simulate human annotation as described in (Plátek and Jurčíček, 2016). We used the DB calls as input features because they showed promising results in (Plátek et al., 2016) for DST. Unfortunately, our model was not able to learn correlation between dialogue history inputs, content of the database and the systems replies. It produces only sophisticated lies mimicking the systems replies and randomly choosing facts from database. Consider example from Figure 5 where both plain encoder-decoder and our propose model decoded the same incorrect reply. Our informal experiments showed that encoder-decoder is able to produce the exact template on DSTC2 dataset in over than 30 % cases without checking the right named entities. Additionally, we find out that over 73 % templates are convenient but they differ only in incorrectly used named entities such telephone number as demonstrated in Figure 5 on small manually checked subset of 100 replies from DSTC2 test set.

We found out that our model learned to ignore the database part of the model and propagated all the information through the encoder-decoder part. We blame mainly huge ambiguity in the data and not enough training examples in the DSTC2 dataset so the model cannot learn the abstraction for representing constraints provided in the history for selecting the right row.

We aim to solve the problem by simplifying the model and learn to predict each row of database

input: *anatolia serves turkish food in the moderate price range what is the phone number and address*
decoded: *The phone number of* **meghna** *is* **01223 727410** *. EOS*
target: *The phone number of anatolia is* **01223 362372** *and it is on 30 Bridge Street City Centre . EOS*

Figure 5: For encoder-decoder framework is easy to learn templates but hard to extract semantic information from the query and the system database. Note, that the decoded reply produced telephone number but of a completely different restaurant (not even the one called **meghna**).

directly, predicting for each row of the DB by a binary classifier if it will appear in the database call results. The advantage is that such architecture allows for multiple possible results which are typical for user queries and it can be easily trained directly using supervised learning if we prepare annotations using simple heuristics or later to be fine-tuned with reinforcement learning. As a side effect, each dialogue history and corresponding system reply training example will be expanded using heuristic functions to many possible replies over possible DB call results. We expect that by explicitly modeling the data ambiguity the trained classifier will generalize better and finally learn to propagate constraints from history over database to results which will be semantically more accurate. We are especially interested in improving the semantic compatibility of the answer which we measure based on the dialogue state annotation from DSTC2 data or DB calls in the dataset (Plátek and Jurčíček, 2016) as recall and accuracy.

5 Future work

Most of the current research focus on reducing handcrafted components from the dialogue system architectures. A successful approach is to built an end-to-end system using neural networks which we introduced in Section 2 and proposed improvements in Section 4.2. However, the most popular approaches require a lot of annotated data. We suggest that a system should collect annotations which are used for its retraining. We aim at reducing both the expert work and annotated data needed for launching a narrow domain task oriented dialogue system. Next, we will not only try to optimize action selection process of a dialogue system, but we will also attempt to learn new facts and thus perform more informed decisions. At the same time, such agent improves its performance by live interactions with users and the annotations may be helpful for adapting to a new domain.

We suggest following experiments to explore the crucial problems which need to be solved before a conversational agent is able to extract information through interaction and later used them for self-improvement. The experiments are designed as a proof of concept on a narrow domain and scaling up to larger or multiple domains are left for future work or as obvious extensions.

Easy first decoding for dialogue state tracking using reinforce algorithm

We show in work (Plátek et al., 2016) that modeling DST as sequence-to-sequence is easy-to-deploy, captures correlation between dialogue state values and also does nut suffer much from data sparsity. However, the order of the dialogue state labels is arbitrary chosen and may not be optimal for prediction dialogue state. In this experiment, we rephrase the DST task as a sequence-to-set model by employing loss function which prefers if three hypotheses labels match the set of gold labels $\{food\_type, area, price\_range\}_{hypotheses} = \{food\_type, area, price\_range\}_{gold}$. Such loss function is in sharp contrast with combined cross-entropy loss maximizing probability true labels in order food, area and price range which is used in sequence-to-sequence approach. The sequence-to-set loss function is not smooth and cannot be differentiated, but we will use reinforce algorithm (Williams, 1992) to update the model parameters. We plan to pre-trained the model with cross-entropy updates and later fine tuned the weights with a reinforce training. Given the same model and the same number of parameters we expect the model trained with the reinforce algorithm to perform better than a cross-entropy training with early stopping. We assume that the reinforce algorithm will benefit from directly optimizing the evaluation function. In addition, it will be interesting to explore which permutation of slots is best to use for DSTC2 dataset and if the best permutation differs for each dialogue. Using the reinforce algorithm on well pre-trained model

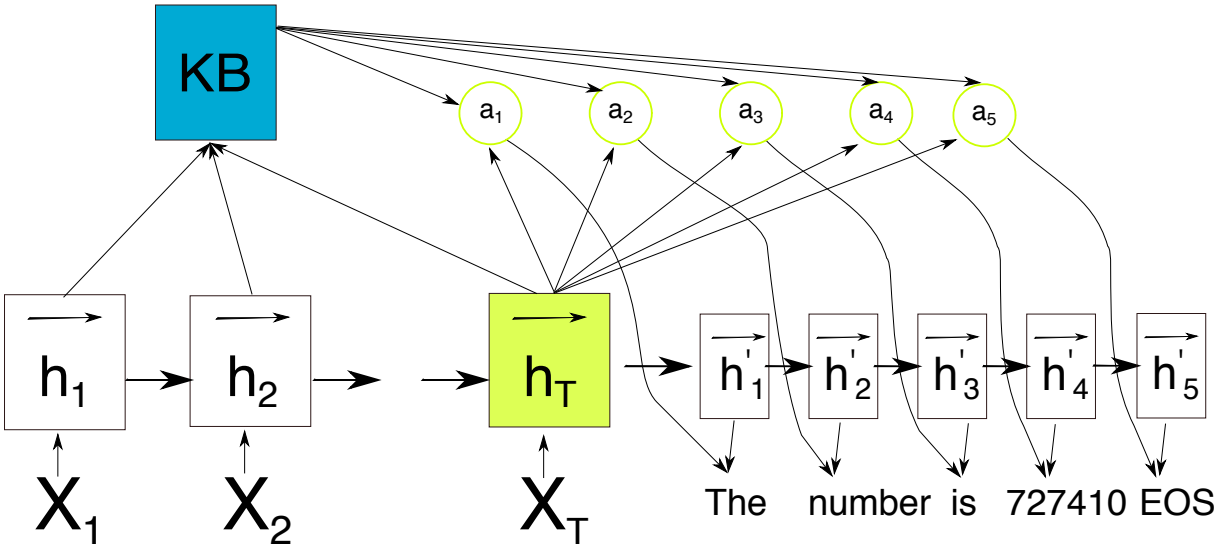

Figure 6: Computational graph of neural network model which demonstrates that conditioning on the database part was completely ignored by the trained model. We hoped that the attention will combine the information from the KB graph and the last hidden state h_T, but it put weight only on the last hidden state h_t for all words as depicted by the yellow colour representing the last hidden state h_t. The output of the KB computation was an weighted embedding of rows of the database similarly as represented in Figure 7.

we want to examine if easy-first decoding of slots can perform better than arbitrary chosen order of predictions.

Discovering the database queries

The dialogue system generates two kinds of actions; responses and calls to its database. The calls to the database influence the system replies in future turns by presenting their results or by changing results of next database calls. This experiment focuses on discovering the database calls which should be used for querying the database from human-human conversations.

Even if the DB results are immediately presented the database operations are not obvious. The first example in Figure 8 demonstrates that the system presents only one of possible results of the DB call to the user. In the second example, the system replies the single possible answer *in the west part* to the query *select restaurant.area where name="India house"*. However, the same result can be obtained by many other queries where the name is replaced by other restaurants; $x = \{travellers\ rest, la\ margherita, \ldots\}$ all of them from west part of the city. As a result, discovering only the queries which makes sense from human perspective is challenging

We formulate the task as a classification problem where the classifier assign high probability to DB calls which are able to produce results compatible with the system reply and ideally the highest probability should be given the DB call intended by the user.

We propose to use the automatic and objective evaluation criteria. Namely, we will check if:

- the system replies with valid property,
- the arguments of the DB call are present in the dialogue history.
- the gold answer is in the result of the DB call.

We will evaluate our classifier on the dataset collected in work (Plátek and Jurčíček, 2016) where if the entity is valid given the history the DB call is unambiguous. In addition, we would like to compare it human judgement on portion of the data.

End-to-end system for consistent DB calls and system replies

In this experiment, we will evaluate the extension of end-to-end system from Section 4.2 which should provide us with reasonable accuracy of selecting the right database call results. We plan to frame the NLG part of the system as classification task of predefined templates for its simplicity. The

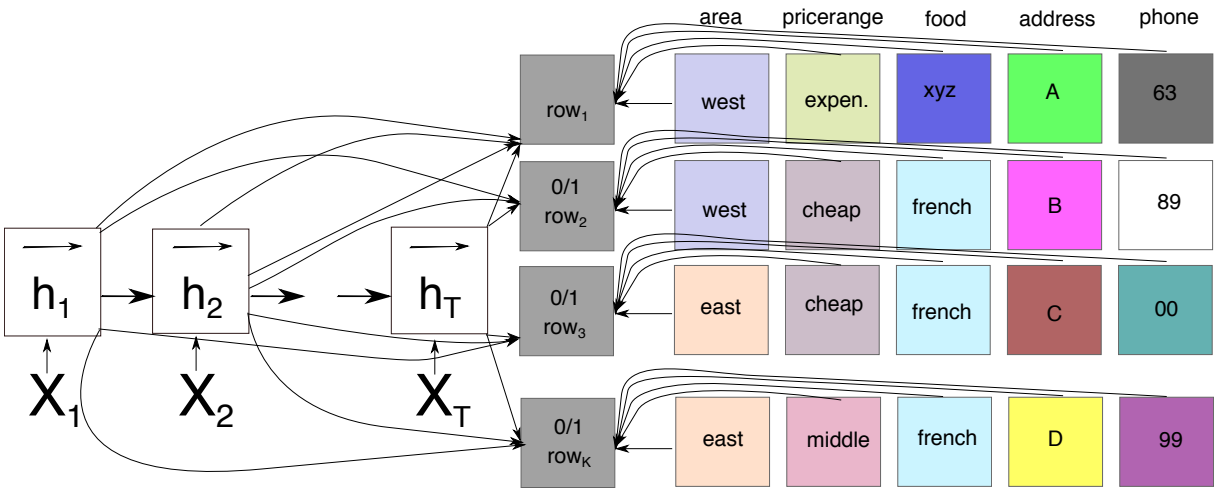

Figure 7: We are currently working on a binary classifiers model which only focus on selecting the right named entities for the reply from database table. Each binary classifier predicts if the restaurant properties stored at corresponding row should be presented in the reply (see the grey column representing the binary decision of the classifier). The dialogue history is simply encoded into series of embeddings x_1, x_2, x_T for each word. The database is represented as set of rows. Note that the input words and the database entities are represented as embeddings which are trained. The colors in the table represents the same embeddings used on different places. The row embeddings are combined embeddings of properties values where all property values are stored in one column. We found out that selecting the right template is quite easy task in comparison to providing the right entities to the template.

...
User: *I would like a Chinese restaurant*
System: *In which area?*
User: *In the city center*
DB: *select restaurant.name where area="city center" and food="Chinese"* System: *A golden house is a Chinese restaurant in the city center*
...
User: *Where is the India house restaurant located?* DB: *select restaurant.area where name="India house"* System: *It is located in the west part of the city.*

Figure 8: Dialogue example with latent calls to system's DB.

overall system (see Figure 9) will predict at each turn the DB call and a template. The template will have placeholders for the DB call arguments and most importantly for the DB call result.[9]

The evaluation will remain the same for choosing the right DB call and its result, the NLG part will be evaluated using accuracy on the golden data from (Plátek and Jurčíček, 2016) dataset and also using consistency with DB call and convenience for the dialogue history. Accuracy evaluated with respect to the golden data is a rather strict criterion, because it may penalize valid responses unseen in the dataset, so we also try to account for other possibilities. Consistency with DB call results, i.e. checking that the arguments of the templates are compatible with golden/predicted DB call, is only necessary requirement for valid template to hold, so it is only a weak measure. Consequently, we plan to validate if a template is convenient reply for given history by crowdsourcing.

[9]Implicit confirmation by repeating the arguments is a common strategy for grounding (Meena and Gustafson, 2014).

Misunderstandings data collection

We assume that error handling is relatively frequent, one should be able to detect it and recover from it (Skantze, 2007). In contrast, to (Skantze, 2007) who focused on ASR errors we focus on following sources of errors which lead to misunderstanding:

- out-of-domain or out-of-application user query and inappropriate system reply,
- ambiguity in the context where one interpretation is intended by the user and the system choose the other one,

- poor action selection of reply or DB call for given history context.

First, we propose to use a special reply for out-of-domain user queries which explains users that his query is out-of-domain and presents system's domain and possibilities. (Plátek and Jurčíček, 2015) With such strategy for handling out-of-domain queries and the errors when system does not inform about its skills, we see the action selection of another reply as an error for given dialogue history. If the misunderstanding is not cause by out-of-domain query the system response may be plainly false for all cases or the user is aware that the system reply might be valid answer, but wants another one.

In this experiment we want to first evaluate how good are we able to detect misunderstanding and second how well are we able to recover from it in the live dialogue. We especially want to focus on the early misunderstanding detection. We will use the dataset prepared in work (Plátek and Jurčíček, 2016) for dialogues without any misunderstanding. Later, we will artificially introduce incorrect system replies to certain dialogue histories and we will collect new continuation of the dialogues after the intentionally introduced nonsensical system reply. We will also ask users to select the most convenient part of the dialogue where to place an out-of-domain question which they are interested about. Again we will collect new follow up conversations and see how humans handle out-of-domain questions about which they do not know any information. We will evaluate accuracy of classifying turns (pair of system and user utterance) into three categories:

- user and system replies are both in the system domain
- user uses out-of-domain utterance
- user attempts to recover from misunderstanding

Optionally, we will investigate if the system uses correct response for recovering from misunderstanding.

We plan to run the experiment as pure Wizard of Oz experiment, and also using a live deployed system.

Data augmentation through exploration

The last of the planned experiments investigates whether our implementation of error recovery detection and our clarification strategies are robust enough to label and thus discover new valid actions for given dialogue history. The idea key idea is that the dialogues tend to be repetitive in narrow domains of task oriented systems. The repetitiveness may not seem natural for human users, and also if one want to train end-to-end systems repetitive data makes the system not only also repetitive too but also less robust.[10] We want to mitigate the repetitiveness of the data by augmenting dialogues by paraphrases.

We will use the original repetitive conversations where the system has reasonable confidence of its action to intentionally slightly change the system reply. If we do not detect misunderstanding in several cases of such new reply for given context, we will suppose we can include it as new data point. The new conversations will be added to original dataset. We suppose that we want to improve an end-to-end systems trained from conversational data such as described at Section 2. The system is able to produce alternatives for each predicted action and also their confidence scores. Using the scores we may not choose the recommended action by the system, but our active learning algorithm will decide on the alternative. We will use coverage and precision of problematic situation described in (Meena, 2016). We plan to use ALI generation and discrimination algorithm described in Section 2.1 which is suitable for overgenerating several candidates. We hope that discriminator will learn to distinguish the responses also on semantic level of the system's domain which current approaches mostly ignore.

Data discovery through misunderstanding

Presenting incorrect information from DB to the users is a worth special attention. Every database is in principle incomplete, and very quickly depreciates if the database contains information like bus stops, street names or telephone numbers which are all examples of entities used in most common task-oriented dialogue systems. Currently, we want to focus only on the incorrect or outdated information and not the missing information. We want to explore if a user is able to de-

[10]The system replies are not only used as targets but also as input features for the system next reply

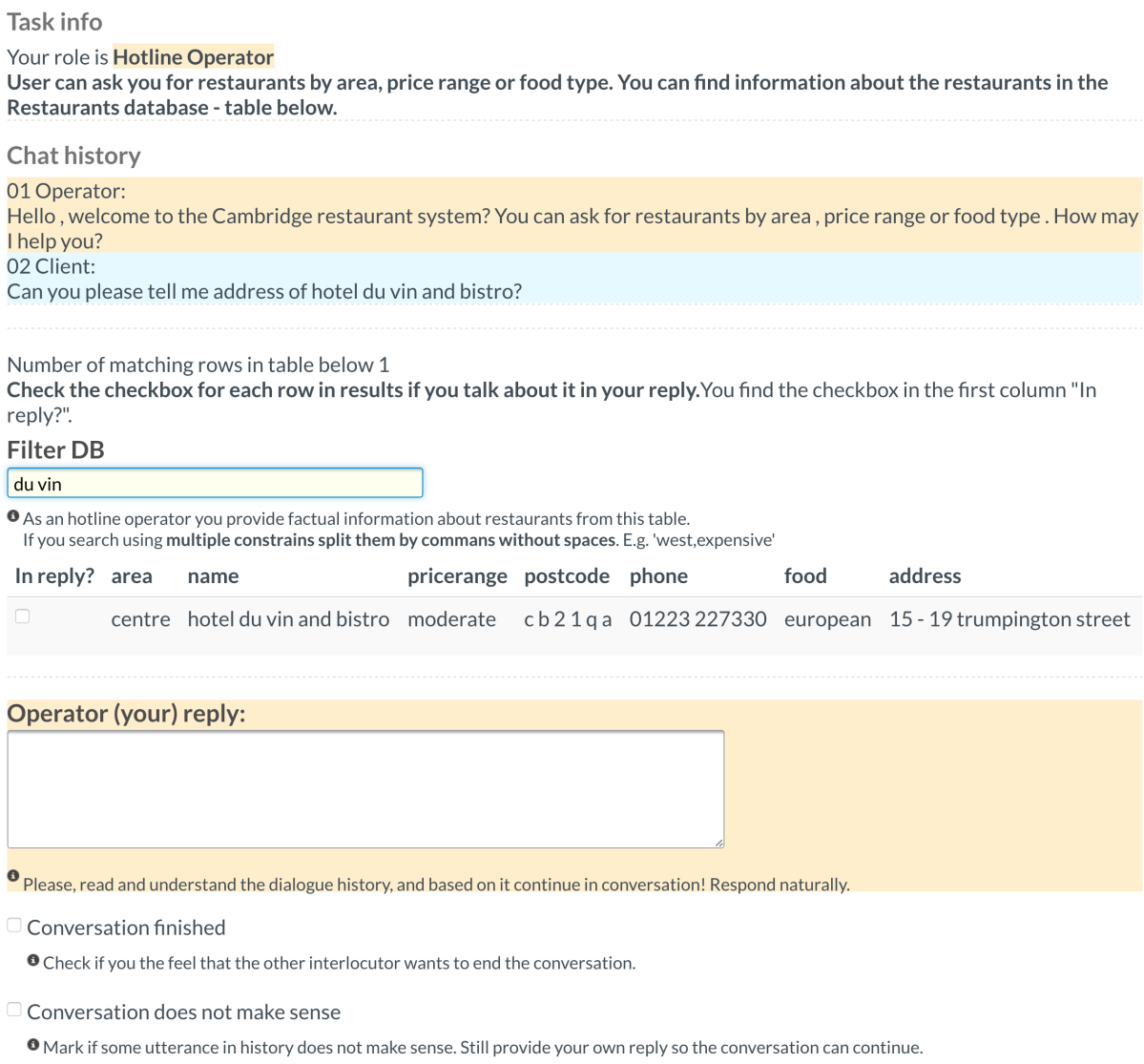

Figure 9: End-to-end system data collection interface

tect that system answer does not match reality and also if it is able to provide the correct answer. In this experiment, we will use crowd-source workers to correct some intentionally outdated information about restaurant domain in Cambridge. The interface displays just the dialogue history for the user and also the DB to the operator (see see Figure 9). For this experiment, a user will not be only informed about restaurant using text system reply which may be outdated, but a leaflet with updated address, price range and menu will be presented to the user. The user will be instructed to tell the system that it is providing incorrect information. We will investigate how often the user notice the mismatch between the information provided and the true state presented in the leaflet. Second, we will evaluate how accurately our system is able to detect that user is correcting its information and how the system is able to parse the information from users answers.

6 Discussion

We see the emerging end-to-end systems (Williams and Zweig, 2016, Dodge et al., 2015, Wen et al., 2016) as a big step forward to reducing expert effort needed in building dialogue systems. However, the human effort put into building even narrow domain task oriented system just shifted from precious expert work to more scalable crowd-source annotators effort (Wen et al., 2016, Serban et al., 2015). We present series of experiments which should study how such scalable approach is pushed even further by collecting annotation through

interactions. In addition, the same methods can be used to extracting and updating knowledge from conversations which is almost unexplored direction how to extend dialogue system domain and knowledge.

The inverse reinforcement learning (IRL) is well established research direction of reducing the need for labeled data (Abbeel and Ng, 2004). The work of (Nouri et al., 2012) uses IRL successfully for a toy task in cultural decision making in negotiations. More recently, (Su et al., 2016) described active reward learning using unsupervised neural network embeddings and Gaussian processes and managed to greatly reduce the amount of data to deploy a system. The problem with active reward learning of particular architecture is that when the single neural network architecture deprecates the new system can use all the knowledge stored in the system parameters in a very limited way. Effectively, it means that only the same starting labeled data together with collected unlabeled data can be used for training new architectures. We see that collecting annotation have bigger benefit from longer point of view and additionally it is completely orthogonal to active reward learning.

The topics of error detection and error recovery of dialogue systems has been described from several points of view, but we learned that most of the experiments are focused on ASR errors (Skantze, 2007). The work of (Meena, 2016) analyzed how to detect and recover also from SLU and DM errors. However, the work of (Meena, 2016) primarily focused to discover errors offline and recommend designers which part of dialogues system needs more attention.

The work of (Pappu, 2014) detects errors over multiple components and then employs post processing step which optimizes the components' pipeline jointly. This attitude is no longer necessary because all our components are optimized jointly. However, the work shows interesting insights on what might be the most common errors and it also uses an error discovery for knowledge acquisition. The authors report promising results especially on acquiring missing named entities and enriching the system knowledge base.

In our work, we want to follow up on their results, integrate our strategies to fully trainable system. The most importantly, we want collect annotations for improving the system itself in addition to learning new facts directly. To our knowledge no other work described a dialogue system which stores user reward signal in explicit form, aka annotations, for its later optimization. We would like to investigate its usefulness and compare it with current approaches such as active reward learning (Su et al., 2016) and zero-shot learning (Vinyals et al., 2016).[11]

7 Conclusion

We presented our work in progress, motivated our past and suggested possible future experiments and discussed their difficulties. Our experiments are mainly described as classification tasks where neural network statistical models will be used. We choose neural networks because they not only efficiently exploit labeled data but also also can be fine-tuned with reinforcement training. Our goal is to develop strategies which will help a conversational agent to collect annotations and facts useful to any statistical inference algorithm.

We realize that we have proposed several research directions, which will be difficult to explore in depth despite their similarities. We plan to continue by exploring the directions in the order they has been presented, and we hope that working on the first problems will teach us what experiment should we explore next.

Acknowledgments

This research was partly funded by the Ministry of Education, Youth and Sports of the Czech Republic under the grant agreement LK11221, core research funding, grant GAUK 1915/2015, and also partially supported by SVV project number 260 333. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40c GPU used for this research. Computational resources were provided by the CESNET LM2015042 and the CERIT Scientific Cloud LM2015085, provided under the program "Projects of Large Research, Development, and Innovations Infrastructures".

References

[Abbeel and Ng2004] Pieter Abbeel and Andrew Y. Ng. 2004. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first in-*

[11] The zero-shot learning applies encoded prior information to extremely efficiently use few data samples provided in the user feedback.

ternational conference on Machine learning, page 1. ACM.

[Bahdanau et al.2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate.

[Bahdanau et al.2016] Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2016. An actor-critic algorithm for sequence prediction.

[Bordes and Weston2016] Antoine Bordes and Jason Weston. 2016. Learning end-to-end goal-oriented dialog.

[Choi and Kim2011] Jaedeug Choi and Kee-Eung Kim. 2011. Inverse reinforcement learning in partially observable environments. 12:691–730.

[Dodge et al.2015] Jesse Dodge, Andreea Gane, Xiang Zhang, Antoine Bordes, Sumit Chopra, Alexander Miller, Arthur Szlam, and Jason Weston. 2015. Evaluating prerequisite qualities for learning end-to-end dialog systems. *CoRR*, abs/1511.06931.

[Dumoulin et al.2016] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron Courville. 2016. Adversarially learned inference.

[Dušek and Jurčíček2016] Ondřej Dušek and Filip Jurčíček. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings.

[Gasic et al.2011] Milica Gasic, Filip Jurčíček, Blaise Thomson, Kai Yu, and Steve Young. 2011. Online policy optimisation of spoken dialogue systems via live interaction with human subjects. In *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*, pages 312–317. IEEE.

[Gers et al.2000] Felix A. Gers, Jürgen Schmidhuber, and Fred Cummins. 2000. Learning to forget: Continual prediction with LSTM. 12(10):2451–2471.

[Henderson et al.2013] Matthew Henderson, Blaise Thomson, and Steve Young. 2013. Deep neural network approach for the dialog state tracking challenge. *Proceedings of the SIGDIAL 2013 Conference*, pages 467–471.

[Henderson et al.2014a] Matthew Henderson, Blaise Thomson, and Jason Williams. 2014a. The second dialog state tracking challenge. In *15th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, volume 263.

[Henderson et al.2014b] Matthew Henderson, Blaise Thomson, and Jason D Williams. 2014b. The third dialog state tracking challenge. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*, pages 324–329. IEEE.

[Henderson et al.2014c] Matthew Henderson, Blaise Thomson, and Steve Young. 2014c. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 292–299.

[Huang et al.1990] Xuedong D. Huang, Yasuo Ariki, and Mervyn A. Jack. 1990. *Hidden Markov models for speech recognition*, volume 2004. Edinburgh university press Edinburgh.

[Krizhevsky et al.2014] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. 2014. *The CIFAR-10 dataset*.

[Liu et al.2016] Chia-Wei Liu, Ryan Lowe, Iulian V. Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation.

[Lowe et al.2015] Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems.

[Lowe et al.2016] Ryan Lowe, Iulian V. Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. On the evaluation of dialogue systems with next utterance classification.

[Mairesse et al.2009] François Mairesse, Milica Gasic, Filip Jurčíček, Simon Keizer, Blaise Thomson, Kai Yu, and Steve Young. 2009. Spoken language understanding from unaligned data using discriminative classification models. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 4749–4752. IEEE.

[Meena and Gustafson2014] Raveesh Meena and Johan Boye Gabriel Skantze Joakim Gustafson. 2014. Crowdsourcing street-level geographic information using a spoken dialogue system. In *15th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 2.

[Meena2016] Raveesh Meena. 2016. Data-driven methods for spoken dialogue systems: Applications in language understanding, turn-taking, error detection, and knowledge acquisition.

[Mikolov et al.2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

[Nouri et al.2012] Elnaz Nouri, Kallirroi Georgila, and David Traum. 2012. A cultural decision-making model for negotiation based on inverse reinforcement learning. In *Proc. of CogSci*.

[Oquab et al.2014] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. 2014. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1717–1724.

[Pappu2014] Aasish Pappu. 2014. Knowledge discovery through spoken dialog.

[Peddinti et al.2015] V. Peddinti, G. Chen, V. Manohar, T. Ko, D. Povey, and S. Khudanpur. 2015. JHU ASpIRE system: Robust LVCSR with TDNNS, iVector adaptation and RNN-LMS. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 539–546.

[Plátek and Jurčíček2015] Ondřej Plátek and Filip Jurčíček. 2015. Self awareness for better common ground. page 200.

[Plátek and Jurčíček2016] Ondřej Plátek and Filip Jurčíček. 2016. Dataset of Operator-client Dialogues Aligned with Database Queries for End-to-end Training.

[Plátek et al.2016] Ondřej Plátek, Petr Bělohlávek, Vojtěch Hudeček, and Filip Jurčíček. 2016. Recurrent neural networks for dialogue state tracking.

[Raux et al.2005] Antoine Raux, Brian Langner, Dan Bohus, Alan W. Black, and Maxine Eskenazi. 2005. Let's go public! taking a spoken dialog system to the real world. In *in Proc. of Interspeech 2005*. Citeseer.

[Serban et al.2015] Iulian V. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015. Building end-to-end dialogue systems using generative hierarchical neural network models.

[Serban et al.2016] Iulian Vlad Serban, Tim Klinger, Gerald Tesauro, Kartik Talamadupula, Bowen Zhou, Yoshua Bengio, and Aaron Courville. 2016. Multiresolution recurrent neural networks: An application to dialogue response generation.

[Skantze2007] Gabriel Skantze. 2007. Error handling in spoken dialogue systems: managing uncertainty, grounding and miscommunication.

[Su et al.2015] Pei-Hao Su, David Vandyke, Milica Gasic, Dongho Kim, Nikola Mrksic, Tsung-Hsien Wen, and Steve Young. 2015. Learning from real users: Rating dialogue success with neural networks for reinforcement learning in spoken dialogue systems. In *INTERSPEECH*.

[Su et al.2016] Pei-Hao Su, Milica Gasic, Nikola Mrksic, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. On-line active reward learning for policy optimisation in spoken dialogue systems.

[Sutskever et al.2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

[Sutton and Barto1998] Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.

[Vejman and Jurčíček2015] Martin Vejman and Filip Jurčíček. 2015. Development of an english public transport information dialogue system.

[Vinyals et al.2016] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning.

[Vodolán et al.2015] Miroslav Vodolán, Rudolf Kadlec, and Jan Kleindienst. 2015. Hybrid dialog state tracker.

[Wen et al.2016] Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, David Vandyke, and Steve Young. 2016. A network-based end-to-end trainable task-oriented dialogue system.

[Wierstra et al.2010] Daan Wierstra, Alexander Förster, Jan Peters, and Jürgen Schmidhuber. 2010. Recurrent policy gradients. 18(5):620–634.

[Williams and Zweig2016] Jason D Williams and Geoffrey Zweig. 2016. End-to-end lstm-based dialog control optimized with supervised and reinforcement learning. *arXiv preprint arXiv:1606.01269*.

[Williams et al.2013] Jason Williams, Antoine Raux, Deepak Ramachandran, and Alan Black. 2013. The Dialog State Tracking Challenge. *Sigdial*, (August):404–413.

[Williams1992] Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. 8(3):229–256.

[Williams2014] Jason D. Williams. 2014. Web-style ranking and SLU combination for dialog state tracking. In *15th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 282.

[Young et al.2010] Steve Young, Milica Gašič, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. 2010. The hidden information state model: A practical framework for POMDP-based spoken dialogue management. *Computer Speech & Language*, 24(2):150–174.

[Yu et al.2016 04] Kai Yu, Lu Chen, Kai Sun, Qizhe Xie, and Su Zhu. 2016-04. Evolvable dialogue state tracking for statistical dialogue management. 10(2):201–215.

[Zhang et al.2016] Yu Zhang, Guoguo Chen, Dong Yu, Kaisheng Yaco, Sanjeev Khudanpur, and James Glass. 2016. Highway long short-term memory RNNS for distant speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5755–5759. IEEE.

[Žilka and Jurčíček2015 07 13] Lukáš Žilka and Filip Jurčíček. 2015-07-13. Incremental LSTM-based Dialog State Tracker.