Domain Transfer in Dialogue Systems without Turn-Level Supervision

Anonymous EMNLP-IJCNLP submission

Abstract

Task oriented dialogue systems rely heavily on specialized dialogue state tracking (DST) modules for dynamically predicting user intent throughout the conversation. State-of-the-art DST models are typically trained in a supervised manner from manual annotations at the turn level. However, these annotations are costly to obtain, which makes it difficult to create accurate dialogue systems for new domains. To address these limitations, we propose a method, based on reinforcement learning, for transferring DST models to new domains without turn-level supervision. Across several domains, our experiments show that this method quickly adapts off-the-shelf models to new domains and performs on par with models trained with turn-level supervision. We also show our method can improve models trained using turn-level supervision by subsequent fine-tuning optimization toward dialog-level rewards.

1 Introduction

Intelligent personal assistants, such as Amazon Alexa, Apple Siri and Google Assistant, are becoming everyday technologies. These assistants can already be used for tasks such as booking a table at your favorite restaurant or the flight for your next vacation. Such dialogue systems potentially allow for smooth interactions with a myriad of online services, but rolling them out to new tasks and domains requires expensive data annotation. In developing goal-oriented dialogue systems, dialogue state tracking (DST) refers to the subtask of incrementally inferring a user's intent as expressed over a sequence of turns. The detected user intent is then used by the dialogue policy in order to decide what action the system should take (Henderson, 2015). For example, in a chatbot-based train reservation system, DST amounts to understanding key information provided by the user as *slot-value pairs*, such as the desired departure and arrival stations, the day and time of travel, among others. With the introduction of the Dialogue State Tracking Challenges (DSTC, Williams et al. (2013)), this line of research has received considerable interest. State-of-the-art models for dialogue state tracking are, as already indicated, learned in a fully supervised setting from datasets where slots and values are annotated manually at the turn level (Mrkšić et al., 2017a; Zhong et al., 2018; Ren et al., 2018; Nouri and Hosseini-Asl, 2018). This allows for high-accuracy models in a select number of domains, where turn-level annotations are available. However, such annotations are cumbersome and costly to obtain, and, in practice, a bottleneck for producing dialogue systems for new domains.

In this paper, we present an approach to DST that pretrains a DST model on a domain for which turn-level annotations exist, e.g., train booking, but fine-tunes to other domains for which no data for turn-level annotations is directly available, e.g., hotel booking. In particular, we use standard maximum likelihood training to induce a supervised model for the source domain, and resort to reinforcement learning (RL) from dialog-level user feedback for transferring to the target domain. This improves target domain performance. In addition to improvements in cross-domain set-ups, we also report small, but consistent gains using dialogue-level feedback to increase in-domain performance.

Contributions To summarize, our contributions are: Relying on *only dialogue-level feedback* for target domain fine-tuning, we show that it is possible to transfer between domains in dialogue state tracking using reinforcement learning, gaining a significant increase in performance over baselines trained using source-domain, turn-level annota-

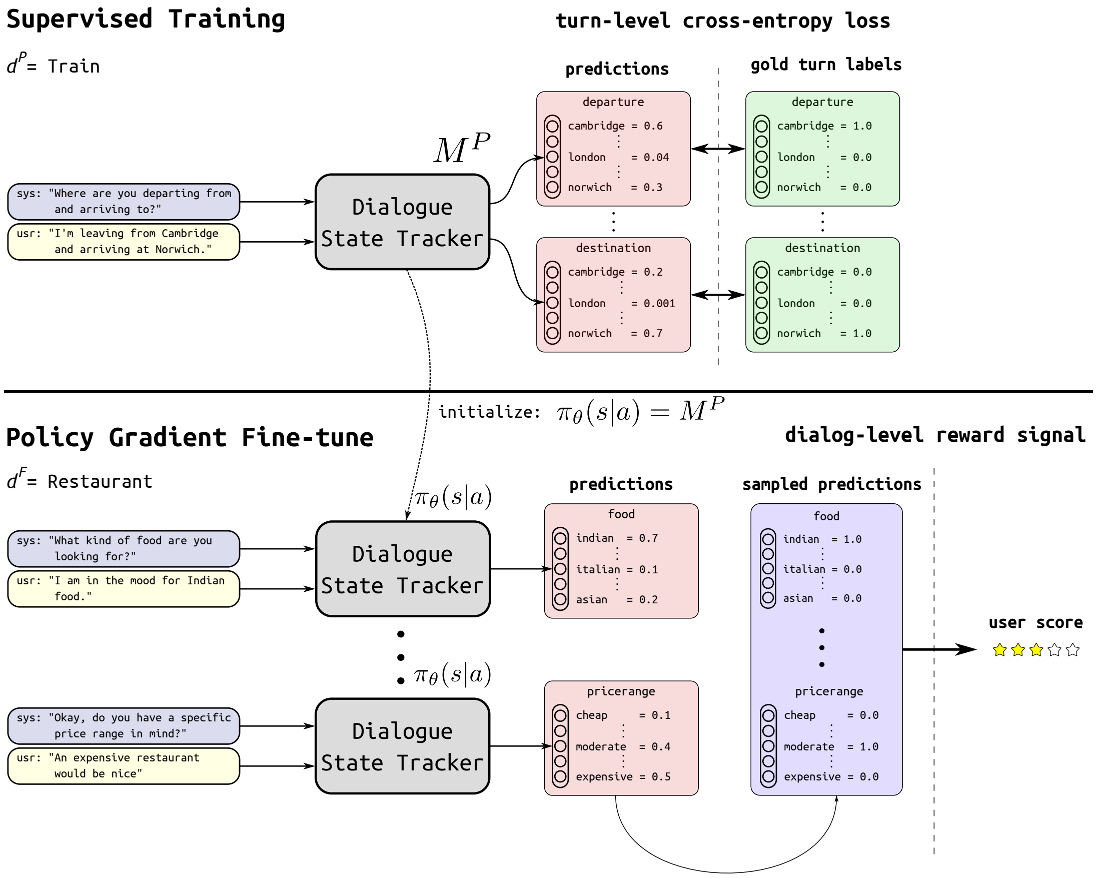

Figure 1: Illustration of our proposed domain transfer dialogue state tracker, using a model M^P trained with turn-level supervision on d^P as a starting point for the fine-tuning policy $\pi_\theta(s|a)$ on domain d^F.

tions. Second, we show that policy gradient methods can also be used to boost the in-domain accuracy of already converged models trained in the usual supervised manner.

2 Baseline Architecture

Our proposed model is based on StateNet (Ren et al., 2018), which uses separate encoders for the two basic inputs that define a turn: the user utterance and the system acts in the previous turn. These inputs are represented as fixed-size vectors that are computed from n-gram based word vector averages, then passed through a number of hidden layers and non-linearities. We concatenate these representations, and, for every candidate slot, we compare the result to slot representations, again derived from word vectors and intermediate layers. We update the hidden state of a GRU encoding the dialogue history.and compare this representation to all candidate values for a given slot. From this, we compute the probability of slot-value pairs. For efficiency reasons, we modify the original StateNet model to only update the GRU that tracks the inner dialogue state after every turn and once all slots are processed within that turn, rather than after every computation of slot values.

Embedding slots and values, and treating them as an input to the model rather than as predefined classes, are important features of StateNet: These features enable zero-shot learning and make the architecture a natural choice for domain transfer experiments, even if it is not the first to enable zero-shot learning in dialogue state tracking in such a way (Zhong et al., 2018; Ramadan et al., 2018). In addition to being well suited for domain transfer, StateNet also produces state-of-the-art results on the DSTC2 and WOZ 2.0 datasets (Henderson et al., 2014a; Mrkšić et al., 2017b).

Training our model is split into two distinct phases. From a pretraining domain d^P for which manual turn-level annotations are available, we learn a model M^P, using the available dialogues to train our system until convergence on a held-out development set. Then, for a further domain $d^F \notin D - d^P$, where D is the set of available domains, we use a policy gradient training to fine-tune M^P to the new domain, based on simulated user feedback, corresponding to how many goals we met at the end of the conversation. Figure 1 presents an overview of this training process.

Pretraining In the pretraining phase, we use our implementation of the StateNet model. Just as Ren et al. (2018), we focus on predicting the user state and use the information about the system acts contained in the data. During pretraining, we rely on turn level supervision, training models on a single domain and evaluating on a held out set from that same domain.

3 Domain Transfer Using Reinforcement Learning

Dialogue state tracking with RL Given a pretrained model M^P trained on a domain d^P, we then seek to fine-tune it on a new domain d^F. Since we do not have turn-level annotations for the target domain, we cannot use maximum likelihood training to adapt to this d^F. This also means that standard domain adaptation methods (Blitzer et al., 2006; Daume III and Marcu, 2006; Jiang and Zhai, 2007) are *not* applicable. Instead, we frame our transfer learning task as a reinforcement learning problem and use policy gradient training. This allows us to use dialogue-level feedback as a reward signal. Policy gradient training has advantages over value-based RL algorithms, including better convergence properties, ability to learn optimal stochastic policies and effectiveness in high-dimensional action spaces (Sutton and Barto, 1998). Within this paradigm, the dialogue state tracker can be seen as an *agent* that interact in the *environment* of a dialogue. Throughout the conversation, the DST model tracks the presence of slots in the conversation and assigns a probability distribution over the values, if present. At the end of a dialogue, represented by a state s, our model goes through the slots and performs an action, a, by sampling a value from the present slot-value probability distribution. It then receives a reward based on how well it predicted slot-value pairs. We illustrate this training regime using dialog-level feedback in the lower half of Figure 1.

Dialog-level reward signal We let our reward simulate the signal the model would receive in a real-world active setting. Dynamically obtaining turn-level feedback is not only costly, but undesirable for the user experience. However, acquiring user feedback at the end of a dialogue is more feasible and common practice in commercial dialogue systems. We now simulate this feedback, which in a real-word application would for instance be given on a 5-star scale, with the ratio of correctly predicted slot-value pairs at the end of the dialogue, assuming a correlation between these factors.

For a set of gold labelled slot-value pairs as the final belief state of the dialog, S_G, and a set of predicted slot-value pairs at the end of the dialog, S_P, we define the reward of the dialogue as their intersection divided by the total amount of gold labels plus the slot-value pairs we predicted wrongly. Our reward thus is defined as

$$R_{goal} = \frac{|S_G \cap S_P|}{|S_G \cup S_P|}$$

which is also referred to as the Jaccard index

Policy Gradient Methods We define the policy network π_θ as the StateNet network, which is initialized with a pretrained model M^P. The weights of the StateNet network are then fine-tuned using stochastic gradient ascent, i.e., in the direction of the gradient of the objective function $\nabla J(\theta)$. The update in the vanilla policy gradient algorithm is:

$$\nabla J(\theta) = \nabla_\theta \log \pi_\theta(a|s) R_{goal} \tag{1}$$

We update the policy of the network after each iteration, following Sutton and Barto (1998).

Variance reduction methods Policy gradient methods suffer form certain shortcomings. For instance, they frequently converge to local, instead of global, optima. Furthermore, the evaluation of a policy is inefficient and suffers from high variance (Sutton and Barto, 1998). A common way to circumvent the above-mentioned issues is to introduce a baseline model (Weaver and Tao, 2001). It is typically initialized as a frozen copy of the pretrained model M^P. The baseline models the reward B_{goal} at the end of the dialog. We can then define an *advantage* of an updated model over the initial one as $A_{goal} = R_{goal} - B_{goal}$. In addition to subtracting the baseline, we also add the entropy $\mathcal{H}(\pi_\theta(a|s))$ of the policy to the gradient to encourage more exploration (Williams and Peng, 1991), in order to counteract the local optima convergence shortcoming. With these modifications to the policy update in Eq. (1), we can rewrite the final gradient as:

Domain	# of dialogues	# of dialogues with only one domain	# of turns (avg.)	# of slots	# of values (processed)	Split sizes (train-dev-test)
TAXI	2057	435	7.66	4	610	326-57-52
TRAIN	4096	345	10.26	6	81	282-30-33
HOTEL	4197	634	10.95	9	187	513-56-67
RESTAURANT	4692	1310	8.78	6	330	1199-50-61
ATTRACTION	3515	150	7.69	2	186	127-11-12

Table 1: Statistics of the MultiWOZ dataset. The reported numbers are from our processed dataset.

$$\nabla J(\theta) = \nabla_\theta \log \pi_\theta(s|a) A_{goal} + \alpha \mathcal{H}(\pi_\theta(s|a)), \quad (2)$$

where α is a term that control influence of the entropy.

Hill climbing with rollbacks Since the policy gradient methods are prone to suffer from performance degradation over time (Kakade, 2002), we employ a rollback method when the policy starts to deviate from the objective. The performance of the model is monitored every few iterations on the development set. If the new model achieves greater rewards than the previously best model, the new model is saved. Contrarily, we roll back to the previous model that performed best and continue from there following other exploration routes if the reward failed to improve for a while. When the policy degrades beyond recovery, the rollback in combination with the slot-value distribution sampling can give a way to a path that leads to greater rewards. We note our hill climbing with rollbacks strategy is an instance of a generalized version of the win-or-learn-fast policy hill climbing framework (Bowling and Veloso, 2001).

4 Experiments

4.1 Data

We use the recently introduced MultiWOZ dataset (Budzianowski et al., 2018) which consists of $10,438$ dialogues spanning 7 domains: ATTRACTION, HOSPITAL, POLICE, HOTEL, RESTAURANT, TAXI and TRAIN. The dataset contains few dialogues in the POLICE and HOSPITAL domains, so we do not include these as the single domain dialogues in these domains did not contain belief state labels. The MultiWOZ dataset consists of natural conversations between a tourist and a clerk from an information center in a touristic city. There are two main types of dialogues. Single-domain dialogues include one domain with a possible booking sub-task. Multi-domain dialogues, on the other hand, include at least two main domains. MultiWOZ is much larger and more complex than other structured dialogue datasets such as WOZ2.0 (Mrkšić et al., 2017b), DSTC2 (Henderson et al., 2014a) and FRAMES (El Asri et al., 2017). In addition, unlike the previous datasets, users can change their intent half way through the conversation, making state tracking much more difficult. Table 1 presents statistics of domains used in experiments with the distinction between the case when the dialogue consists of only one or more domains.

Preprocessing MultiWOZ The user utterances and system utterances used to trained our models contain tokens that were randomly created during the creation of the data to simulate reference numbers, train ids, phone numbers, arrival and departure times and post codes. We delexicalize all utterances by replacing these randomly generated values with a special generic token. In addition, we replace the turn label values with this special token and add that to the ontology. As mentioned by Mrkšić et al. (2017a), delexicalizing all values is not scalable to large domains as that requires to always have a dictionary holding all possible values. Therefore, we do not delexicalize any other values. As MultiWOZ is formatted differently than most state tracking datasets, we convert the data into a similar format as WOZ and WOZ2.0, which contain turn labels. Since MultiWOZ only contains the current belief state at each turn, we create the labels by registering the changes in the belief state from one turn to the next. The annotators were given instructions on specific goals to follow, however at times they did not follow this goal. This lead to errors in the belief state such as wrong labels or missing information. These instances also propagate further down to our assigned gold turn labels. Furthermore, while preprocessing the data, we found that there are more values present than reported on the ontology, therefore the number of values presented here is higher than what is reported in Budzianowski et al. (2018). We release our preprocessed data and preprocessing scripts.[1]

4.2 Implementation Details

Our pretrained StateNet model is implemented without parameter sharing and is not initialized with single-slot pretraining as in (Ren et al., 2018). We use the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 10^{-3}. We use an n-gram utterance representation size of 3 and 3 multi-scale receptors per n-gram. The supervised models are trained using a batch size of 16. The size of the GRUs hidden state is 200 and the size of the word embeddings is 400. In line with recent methods for dialogue state tracking, we use fixed pretrained embeddings and do not update them during the training (Mrkšić et al., 2017a; Ren et al., 2018; Zhong et al., 2018). We use the established splits for train, development and testing and apply early stopping if the joint goal accuracy has not improved over 20 epochs.

When fine-tuning with policy gradient, we evaluate on the development set every 5 batches, saving the model if the reward has increased since last. We use an independent hill climbing patience factor of 15, reverting back to the previous best model if no improvements were made in that period. We use a batch size of 16 in our fine-tuning experiments. When applying policy gradient methods in practice, larger batch sizes have shown to lead to more accurate policy updates (Papini et al., 2017), but due to the relatively small training sets we found a batch size of 16 gave us the best sample efficiency trade-off.

Our implementation uses the PyTorch framework (Paszke et al., 2017) and is made publicly available.[2]

4.3 Experimental Protocol

Setups In our experiments, we report a number of different results: 1) Training a DST model M^P with the usual turn-level supervision on the different domains. We only use dialogues which strictly contains the labels of that single domain. We hypothesize that this serves as an upper bound to the performance of the policy gradient fine-tuning.

[1]Link anonymized for the review period.

[2]Link anonymized for the review period.

2) Evaluating the pretrained models as a cross-domain zero-shot baseline. We take a model pretrained on d^P and measure its performance on d^F for all domains in $D - d^P$. This serves as the lower bound for the performance of the policy gradient fine-tuned models. We use this baseline and not a model fine-tuned on d^F with cross entropy training with dialogue level supervision on the final belief state, as we simulate not having gold labels for each slot-value pair, but rather only a scalar rating as the sole signal. 3) Fine-tuning the pretrained model M^P to all other domains with policy gradient as described in Section 3. We experiment with domain transfer from d^P to all domains in $D - d^P$ using only the user simulated dialog-level reward using policy gradient. 4) Lastly, we report the results of fine-tuning a model using policy gradient on the same domain it was pretrained on, d^P, after convergence in order to see if the dialog-level reward signal can further improve its performance. We here use the same training and development data as the supervised model was trained on.

Metric We measure the performance of our models with what we refer to as the *turn level accuracy* metric, which measures the ratio of how many of the gold turn labels are predicted by the DST model at each turn. The reported accuracy is the mean of all turns in the evaluation set.

5 Results

In Table 2 we present the results from our baseline StateNet model and from policy gradient training for the in- and out-of domain scenarios. We also report the average out-of-domain accuracies for each domain, to illustrate how policy gradient training in general performs compared to the baseline. The table show the performance of transferring from each domain to all other domains. From the results we observe that in almost all domain transfer settings, with the exception of RESTAURANT to ATTRACTION, we get a consistent increase in performance when applying policy gradient fine-tuning, compared to the zero-shot transfer baselines. In some instances we also see an increase in performance from further fine-tuning a model after turn-level supervision convergence using only the dialogue-level reward feedback. In the case of ATTRACTION, we are even able to increase the accuracy by a large margin using in-domain policy gradient fine-tuning. On average, we see relative improvements of the accu-

Pretrain \ Finetune	TAXI		TRAIN		HOTEL		RESTAURANT		ATTRACTION	
	BL	PG	BL	PG	BL	PG	BL	PG	BL	PG
TAXI	0.35	0.35	0.17	**0.27**	0.04	**0.10**	0.12	**0.29**	0.00	**0.11**
TRAIN	0.13	0.13	0.43	0.43	0.07	**0.08**	0.08	**0.22**	0.00	0.00
HOTEL	0.004	**0.26**	0.02	**0.19**	0.30	**0.33**	0.10	**0.19**	0.06	**0.11**
RESTAURANT	0.04	**0.25**	0.13	**0.27**	0.11	**0.13**	0.33	**0.34**	**0.11**	0.05
ATTRACTION	0.00	**0.27**	0.00	**0.39**	0.00	**0.08**	0.05	**0.10**	0.11	**0.17**
AVERAGES	0.04	**0.23**	0.08	**0.28**	0.06	**0.10**	0.09	**0.2**	0.04	**0.07**

Table 2: Fine-tuning results for our pretrained baseline (BL) and the policy gradient (PG). The colored results along the left-to-right downward diagonal are in-domain results, dark red being the supervised results and light green the policy gradient fine-tuned results, and each pair of columns compare baseline and system results for each target domain. The AVERAGES rows presents the average out-of-domain transfer scores for each domain.

racy, ranging from 0.03 to 0.2, when applying our proposed method of fine-tuning for DST domain transfer.

6 Analysis

In order to illustrate the effectiveness of doing PG fine-tuning compared to doing zero-shot domain transfer, we plot in Figure 2 the results of training a model on the source domain HOTEL while evaluating, on the development set, its zero-shot accuracy on the target domain TAXI, until convergence on the source domain. After convergence we show how the PG fine-tuning uses the pretrained model as a starting point to further improve the accuracy on the target domain using only the dialog-level feedback. Figure 2 also illustrates the importance of the hill climbing technique we employ. When the performance starts to deteriorate, it manages to revert back to a reasonable baseline and improve performance from there instead. From the blue baseline curve, we also observe that even though the accuracy continuously improves on the source domain, this is not necessarily an indication of the performance on the target domain. On the contrary, performance suddenly starts to deteriorate for the latter when the model overfits to the source domain.

6.1 Error Analysis

In general we observe lower scores for both the baseline models and in-domain fine-tuning on the ATTRACTION domain. We believe this can be attributed to the fact that it only contains 150 dialogues, leaving very little data for the development and test splits. Coupled with the fact that it has 2 slots and 180 values, the risk of encountering unseen slot-value pairs increases significantly.

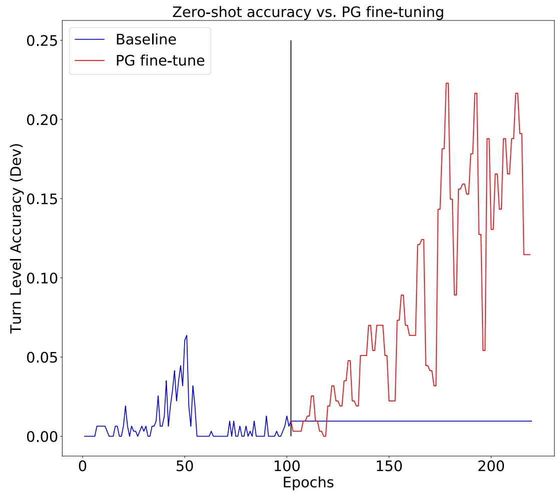

Figure 2: The performance of the supervised model trained on the HOTEL domain while evaluated on the development set of the TAXI domain after each epoch until convergence on HOTEL versus the improvements we get from the policy gradient fine-tuning using the supervised model as starting point.

In Table 3 we present a couple of example turns from the test set of the RESTAURANT domain, with the system utterance, user utterance and the predicted slot-value pairs for both the baseline model, which has been trained on the HOTEL domain, and the PG fine-tuned model. The slot-value pairs in green show correct predictions, whereas pairs in red show incorrect predictions. From the predicted slot-value pairs, we can for example see how the fine-tuned model to a better extent is able to utilize the user and system utterances to correctly predict what price range the user is looking for, even though the baseline correctly predicts the slot presence.

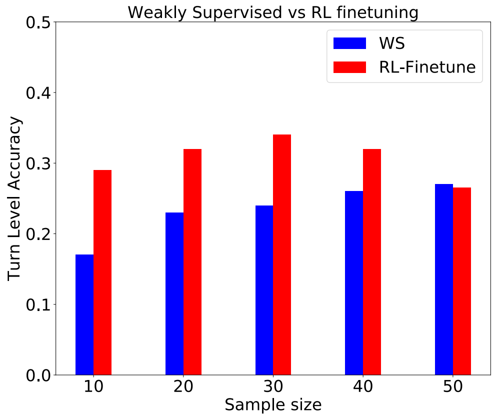

Figure 3: The turn level accuracy of our weakly supervised fine-tuning compared to fine-tuning using PG. Performance plateaus after about 50 samples for both methods.

6.2 Comparisons to Weak Supervision

We also pose the question of how many annotated dialogues in the target domain are needed before policy gradient fine-tuning with dialogue-level rewards is no longer beneficial, compared to fine-tuning a model trained with turn-level cross entropy. In order to further investigate this, we use our pretrained model in the TAXI domain and further finetune with varying amounts of dialogues i.e. $s \in [10, 20, 30, 40, 50]$ using turn level supervision for the RESTAURANT domain. We then fine-tuned each of the models on the RESTAURANT domain using the dialogue-level reward only. The results for these experiments are shown in Figure 3. Overall, we find that when we annotate just 10 complete dialogues and then fine-tune our model using reinforcement learning we still see an increase in performance. We observe that as we increase the sample size s for our weakly supervised models, fine-tuning using policy gradient comes with diminishing returns. At around 50 samples, the performance of the weakly supervised baseline reaches the performance of our system, and improvements from reinforcement learning, if any, become significantly smaller.

7 Related Work

DST architectures The goal of Dialogue State Tracking is to predict the user intent or *belief state* at each turn of the conversation. The range of user goals or, *slots* and *value* pairs, that can possibly be recognized by the system are contained in the domain ontology. DST has for long been a part of spoken dialogue systems, however, before the Dialogue State Tracking challenges (Williams et al., 2013; Henderson et al., 2014a) many of the early architectures relied on hand crafted rules (Wang and Lemon, 2013; Sun et al., 2014, 2016).

Later research has proposed RNN models that exploit delexicalized features (Henderson et al., 2014b; Mrkšić et al., 2015; Rastogi et al., 2017) in order to allow the model to perform better and achieve generalization by reducing the amount of labels. Delexicalization requires that all possible mentions of a slot and value are contained in a lexicon which does not become scalable in larger domains. To address this, Mrkšić et al. (2017a) proposed a neural belief tracker which uses pretrained word embeddings to represent user utterances, system acts and current candidate slot-value pairs and utilizes these as inputs into a neural network.

Recent approaches have proposed sharing parameters across estimators for the slot-value pairs (Zhong et al., 2018; Ren et al., 2018; Ramadan et al., 2018; Nouri and Hosseini-Asl, 2018). Although not extensively investigated, this would make the model more scalable as the amount of parameters would not increase while the ontology size grows. In our experiments, we adopt the model by Ren et al. (2018) as our supervised baseline.

Domain transfer A key issue that remains unexplored by many of the existing methods within DST is domain adaptation. Williams (2013) presented some of the earliest work dealing with multi-domain dialogue state tracking, investigating domain transfer in two dimensions: 1) sharing parameters across slots, 2) sharing parameters across single domain systems. Later research further expanded by using disparate data sources in order to train a general multi-domain belief tracker (Mrkšić et al., 2015). The tracker is then fine-tuned to a single domain to create a specialized system that has background knowledge across various domains. Furthermore, Rastogi et al. (2017) proposed a multi-domain dialogue state tracker that uses a bidirectional GRU to encode utterances from user and system which are then passed in combination with candidate slots and values to a feed-forward network. Unlike our proposed method, they rely on delexicalization of all values.

System utterance	User utterance	Baseline prediction	PG fine-tune prediction
N/A	I'm looking for a cheap place to dine, preferably in the centre of town.	inform(area=center) inform(pricerange=expensive)	inform(area=center) inform(pricerange=cheap)
Yes, I have 4 results matching your request, is there a price range you're looking for?	I would like moderate price range please.	inform(pricerange=expensive)	inform(pricerange=moderate)
There are a number of options for Indian restaurants in the centre of town.What price range would you like ?	I would prefer cheap restaurants.	inform(pricerange=expensive)	inform(pricerange=cheap)

Table 3: Comparison of example turn predictions from the MultiWOZ dataset between the baseline model trained on the HOTEL domains, and the policy gradient fine-tuned model. Green indicates a correct prediction whereas red indicates a wrong prediction.

In addition, their GRU shares parameters across domains. Ramadan et al. (2018) introduced an approach which leverages the semantic similarities between the user utterances and the terms contained in the ontology. In their proposed model, domain tracking is learned jointly with the belief state following Mrkšić and Vulić (2018).

We want to emphasize that all previous models assume the existence of dialogue data annotated at the turn level in the new domain. In our proposed method, we model a more realistic scenario in which we only have a score of how accurate the system was at the end of the dialogue given the final user goal.

Reinforcement Learning in Dialogue In task-oriented dialogues, the reinforcement learning framework has mostly been used to tackle dialogue policy learning (Singh et al., 2002; Williams and Young, 2007; Li et al., 2009; Liu et al., 2018). Gasic et al. (2013) proposed a method to expand a domain to include previously unseen slots using Gaussian process POMDP optimization. While they discuss the potential of their model in adapting to new domains, their study does not present results in multi-domain dialogue management.

Recent work has attempted to build end-to-end systems that can learn both user states and dialogue policy using reinforcement learning. Zhao and Eskenazi (2016) propose an end-to-end dialogue model that uses RL to jointly learn state tracking and dialogue policy. This model augments the output action space with predefined API calls which modify a query hypothesis which can only hold one slot value pair at a time. Dhingra et al. (2017) instead show that providing the model with the posterior distribution of the user goal over a knowledge base, and integrating that with RL, leads to higher task success rate and reward.

In contrast to our work, Gašić et al. (2017) have tackled the problem of domain adaptation using RL to learn generic policies and derive domain specific policies. In a similar study, Chen et al. (2018) approach the problem of domain adaptation by introducing slot-dependent and slot-independent agents.

Our approach differs from the previously presented models in several ways: a) we track the user state using RL, however, we do not learn generic and specific policies ; b) we use RL to adapt models across many domains and a large number of *slot,value* pairs; and c) we assume that a reward is only known for target domain dialogues at the end of each dialogue.

8 Conclusion

This paper tackles the challenge of transferring dialogue state tracking models across domains without having target-domain supervision at the turn level; that is, without manual annotations, which are costly to obtain. Our setup is motivated by the fact that in a practical setting it is much more feasible to obtain dialogue level signals. We introduce a transfer learning method to address this, using supervised learning to learn a base model and then using reinforcement learning for fine-tuning using our dialogue level reward. Our results show consistent improvements over domain transfer baselines without fine-tuning, at times showing similar performance to in-domain models. In addition to demonstrating our models' potential in domain transfer, we show that using the dialogue-level reward signal for fine-tuning can further improve in-domain performance.

References

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of EMNLP*.

Michael Bowling and Manuela Veloso. 2001. Rational and convergent learning in stochastic games. In *IJCAI*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. MultiWOZ-A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Lu Chen, Cheng Chang, Zhi Chen, Bowen Tan, Milica Gašić, and Kai Yu. 2018. Policy adaptation for deep reinforcement learning-based dialogue management. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6074–6078. IEEE.

Hal Daume III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126.

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. Towards end-to-end reinforcement learning of dialogue agents for information access. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 484–495.

Layla El Asri, Hannes Schulz, Shikhar Sharma, Jeremie Zumer, Justin Harris, Emery Fine, Rahul Mehrotra, and Kaheer Suleman. 2017. Frames: a corpus for adding memory to goal-oriented dialogue systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 207–219.

Milica Gasic, Catherine Breslin, Matthew Henderson, Dongho Kim, Martin Szummer, Blaise Thomson, Pirros Tsiakoulis, and Steve Young. 2013. Pomdp-based dialogue manager adaptation to extended domains. In *Proceedings of the SIGDIAL 2013 Conference*, pages 214–222.

Milica Gašić, Nikola Mrkšić, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2017. Dialogue manager domain adaptation using gaussian process reinforcement learning. *Computer Speech & Language*, 45:552–569.

Matthew Henderson. 2015. Machine learning for dialog state tracking: A review. In *Proc. of The First International Workshop on Machine Learning in Spoken Language Processing*.

Matthew Henderson, Blaise Thomson, and Jason D Williams. 2014a. The second dialog state tracking challenge. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 263–272.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014b. 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.

Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *Proceedings of ACL*.

Sham M Kakade. 2002. A natural policy gradient. In *Advances in neural information processing systems*, pages 1531–1538.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *ICLR*.

Lihong Li, Jason D. Williams, and Suhrid Balakrishnan. 2009. Reinforcement learning for dialog management using least-squares policy iteration and fast feature selection. In *INTERSPEECH*.

Bing Liu, Gokhan Tür, Dilek Hakkani-Tür, Pararth Shah, and Larry Heck. 2018. Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 2060–2069.

N Mrkšić, DO Séaghdha, B Thomson, M Gašić, PH Su, D Vandyke, TH Wen, and S Young. 2015. Multi-domain dialog state tracking using recurrent neural networks. In *ACL-IJCNLP 2015-53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference*, volume 2, pages 794–799.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017a. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1777–1788.

Nikola Mrkšić and Ivan Vulić. 2018. Fully statistical neural belief tracking. In *Proceedings of ACL*, pages 108–113.

Nikola Mrkšić, Ivan Vulić, Diarmuid Ó Séaghdha, Ira Leviant, Roi Reichart, Milica Gašić, Anna Korhonen, and Steve Young. 2017b. Semantic specialization of distributional word vector spaces using monolingual and cross-lingual constraints. *Transactions of the Association of Computational Linguistics*, 5(1):309–324.

Elnaz Nouri and Ehsan Hosseini-Asl. 2018. Toward scalable neural dialogue state tracking. In *NeurIPS 2018, 2nd Conversational AI workshop*.

Matteo Papini, Matteo Pirotta, and Marcello Restelli. 2017. Adaptive batch size for safe policy gradients. In *Advances in Neural Information Processing Systems*, pages 3591–3600.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

Osman Ramadan, Paweł Budzianowski, and Milica Gasic. 2018. Large-scale multi-domain belief tracking with knowledge sharing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 432–437.

Abhinav Rastogi, Dilek Hakkani-Tür, and Larry Heck. 2017. Scalable multi-domain dialogue state tracking. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 561–568. IEEE.

Liliang Ren, Kaige Xie, Lu Chen, and Kai Yu. 2018. Towards universal dialogue state tracking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2780–2786.

Satinder Singh, Diane Litman, Michael Kearns, and Marilyn Walker. 2002. Optimizing dialogue management with reinforcement learning: Experiments with the njfun system. *Journal of Artificial Intelligence Research*, 16:105–133.

Kai Sun, Lu Chen, Su Zhu, and Kai Yu. 2014. A generalized rule based tracker for dialogue state tracking. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 330–335. IEEE.

Kai Sun, Su Zhu, Lu Chen, Siqiu Yao, Xueyang Wu, and Kai Yu. 2016. Hybrid dialogue state tracking for real world human-to-human dialogues. In *INTERSPEECH*, pages 2060–2064.

Richard S. Sutton and Andrew G. Barto. 1998. *Introduction to Reinforcement Learning*, 1st edition. MIT Press, Cambridge, MA, USA.

Zhuoran Wang and Oliver Lemon. 2013. 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*, pages 423–432.

Lex Weaver and Nigel Tao. 2001. The optimal reward baseline for gradient-based reinforcement learning. In *UAI*.

Jason Williams. 2013. Multi-domain learning and generalization in dialog state tracking. In *Proceedings of the SIGDIAL 2013 Conference*, pages 433–441. Association for Computational Linguistics.

Jason Williams, Antoine Raux, Deepak Ramachandran, and Alan Black. 2013. The dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*, pages 404–413.

Jason D. Williams and Steve Young. 2007. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393 – 422.

Ronald J Williams and Jing Peng. 1991. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268.

Tiancheng Zhao and Maxine Eskenazi. 2016. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. pages 1–10.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1458–1467.