Cross-Domain Training for Goal-Oriented Conversational Agents

Alexandra Bodirlau, Stefania Budulan, Traian Rebedea
University Politehnica of Bucharest, Romania
alexandra.bodirlau@stud.acs.upb.ro
{stefania.budulan, traian.rebedea}@cs.pub.ro

Abstract

Goal-Oriented Chatbots in fields such as customer support, providing specific information or general help with bookings or reservations, suffer from low performance partly due to the difficulty of obtaining large domain-specific annotated datasets. Given that the problem is closely related to the domain of the conversational agent and that data belonging to a specific domain is difficult to annotate, there have been some attempts at surpassing these challenges such as unsupervised pre-training or transfer learning between different domains. A more thorough analysis of the transfer learning mechanism is justified by the significant boost of the results demonstrated in the results section. We describe extensive experiments using transfer learning and warm-starting techniques with improvements of more than 5% in relative percentage of success rate in the majority of cases, and up to 10x faster convergence as opposed to training the system without them.

1 Introduction

Goal-Oriented Conversational Agents (GO Chatbots) are seeing increased use to help users to achieve predetermined goals, but they can handle only very simple tasks, such as playing songs, searching information, set alarms or reminders. Building a dialogue agent to fulfill complex tasks remains one of the fundamental challenges for the Natural Language Processing (NLP) community and Artificial Intelligence (AI) in general.

There are two dominant approaches for solving this problem. The first one relies on (fully) supervised learning, e.g. using sequence-to-sequence (Sutskever et al., 2014) models, encoding a user's utterance and its context to decode the answer provided by the chatbot. However, this method does not explicitly allow to locate and make use of specific information such as entity recognition (e.g. a person's workplace) and requires large amounts of data in order to flawlessly extract and process particular pieces of information relevant for the task at hand, usually a mandatory requirement for GO Chatbots.

The second category entails partitioning the dialog system into smaller subsystems, usually implemented and trained separately. An example of such a system (Li et al., 2017) consists of several components: Natural Language Understanding, Dialog Manager, and Natural Language Generation. The Dialog Manager is often implemented with the aid of reinforcement learning (RL) based techniques, for instance using Deep Q-Nets (DQN) (Mnih et al., 2015) and having the main goal of learning the policy on account of which the agent will be able to provide answers.

The first approach is used with more favourable outcomes in the case of open-domain dialogue systems (Serban et al., 2016) than in closed-domain dialogue systems (Peng et al., 2017), because it does not require a method to reward the accomplishment of the task. Instead, the success of the conversation resides in the engagement of the user, measured in the level of coherence and cohesion of the dialog. The second approach better fits learning tasks having less labeled data, where the validity of the answer can be determined through evaluating the task's completion (e.g. making a restaurant reservation). These RL-based dialogue systems have the ability to simulate conversations, thus exploring the unknown dialogue space efficiently.

Currently reduced performance of domain-specific conversational agents in fields such as cus-

Proceedings of Recent Advances in Natural Language Processing, pages 142–150,
Varna, Bulgaria, Sep 2–4, 2019.
https://doi.org/10.26615/978-954-452-056-4_017

tomer support, providing certain information or general help with reservations etc., is partly due to the difficulty of obtaining large annotated datasets. With each new domain and each new conversation flow introduced by a new task, newly annotated data need to be fed into the system in order to assimilate them and later provide the best answer for different inputs. The efforts for selecting, categorising and annotating the data are substantial, no matter the previous experience or domain-knowledge. This paper analyses the possibility to alleviate the data annotation endeavor through the inter-domain transfer learning technique (Ilievski et al., 2018). Alongside with unsupervised pre-training and others, transfer learning has proven a significant contribution to deliver better results, but it has only been tested with datasets from a small number of domains. We experiment with larger datasets, wider scenarios and we offer a richer understanding of the method, premises and results for overcoming the lack of data.

In this paper, we provide a thorough study on the impact of transfer learning in goal-oriented chatbots, starting from the work presented by (Ilievski et al., 2018). They proposed the possibility to reuse the knowledge gained from a source domain to boost the training and testing performance of a machine learning chatbot model on a different target domain, as described in more detail in the following sections. They identify two cases:

- **domain overlap** - the source and target domains are different, but share a fraction of actions, and
- **domain extension** - the source domain is extended by the target domain.

In both cases, there are common actions that justify the transfer learning between domains instead of independently training models for each of them. This approach has two effects: (1) the success rate of the model obtained with transfer learning is significantly higher than that of the model trained without any prior knowledge, and (2) transfer learning can be an alternative or complementary to warm starting, which also requires labeled data.

The results presented by the aforementioned authors represent a significant improvement for GO Chatbots, but they are obtained with only three domains, with relatively small datasets: Movie Booking, Restaurant Booking, and Tourist Info. In order to train a model for a more complex domain such as customer support, the improvement has to be validated on multiple datasets from different domains. Also, because the cost of annotating data for such a domain is very high, transfer learning methods should be studied for possible improvements that increase the automation of domain-specific conversations with few data.

The rest of the paper is organized as follows. Section 2 presents the related work for Goal Oriented Chatbots. The model used in our experiments is detailed in Section 3 and the datasets in Section 4. The results of our experiments are described in detail and interpreted in Section 5. Finally, future improvements and conclusions are presented in Section 6.

2 Related Work

We have already classified the existing solutions used for building machine learning chatbots in two categories, based on the learning method: supervised learning and reinforcement learning. In this section, we are providing a more in depth analysis of these two alternatives.

Serban et al. (2016) propose a solution for non-goal-driven systems, which uses an encoder-decoder model and word embeddings to generate the response of the agent starting from the utterance of the user as input. The architecture is composed from two RNNs: one for the utterance level, which treats the dialogue as a sequence of utterances, and one at the word level, which processes an utterance as a sequence of words. This model is trained on movie scripts and the dialogues include the speech acts. A detail worth mentioning here is that the pre-training is performed on a large related, but non-dialogue, corpus. The consequence is that the model accomplishes slightly better results compared with an initialization with fixed word embeddings.

Another supervised learning model for chatbots is presented by (Wen et al., 2017). The architecture is significantly more complex, and is divided in several modules. The utterances received from the user are converted into two representations: a probability distribution over the slot-value pairs called the belief state, and an intent representation generated by an intent network. A database operator selects the most probable values in the belief state and makes a query to the database. A policy network takes as input the intent representation,

database result, and belief state and returns a representation of the next system action. Finally, a generation network uses the action representation to generate a template sequence, which is filled with actual values from the database. This system is very similar to the one used in the current paper, but the former is trained in a supervised fashion and, therefore, it is possible to fail at finding a good policy due to the shortcomings in dialogue exploration. Firstly, the policy is learned by a network instead of using RL (our case) and, secondly, the $\epsilon - greedy$ policy used in our experiments ensures the exploration of unknown states, instead of relying entirely on seen training data and rigid choices.

A possible solution for the disadvantage of using supervised training in the model presented above is proposed by Su et al. (2016). The architecture is similar, but there is a difference in the policy network training: it receives the current state and predicts the next system action in a supervised fashion in the first phase, followed by a reinforcement learning phase. The purpose of the second phase is to improve the generalization capacity of the policy by a better exploration of the action space using reinforcement learning.

A step forward in solving complex tasks is done by Peng et al. (2017). They introduce a hierarchical deep reinforcement learning architecture for solving composite tasks for travel planning, that are a collection of subtasks such as: book air ticket, reserve hotel room, buy train ticket, etc. This type of tasks are a challenge for RL approaches because of the reward sparsity, the slot constraints between different subtasks and the agent's tendency to switch between different subtasks frequently when conversing with users, which leads to poor user experience. The dialogue manager consists of (1) a top-level dialogue policy that selects subtasks, (2) a low-level dialogue policy that selects the actions in a given subtask, and (3) a global state tracker that supervises the cross-subtask constraints.

Ilievski et al. (2018) use the transfer learning mechanism for chatbots employing neural models to reduce the amount of training data and speed up the learning process for new domains. This can be accomplished with the transfer of the parameters learned in a source domain to a target domain, which has some common actions with the former. In order to apply the transfer, it is necessary to have the same state distribution in both domains, therefore the bots trained on the source domain must be aware of the actions in the target domain. They obtain an improvement of 65% in success rate in the case of domain extension and 20% for domain overlap. This represents a noteworthy result and one of the reasons we chose to study this mechanism in more detail. Another reason is the faster learning resulted from the combination of transfer learning with warm start.

In a more recent paper, Wolf et al. (2019) present the improvement brought by transfer learning in generative tasks such as open-domain dialog generation. A Transformer model (Vaswani et al., 2017) is pre-trained on a large unlabeled dataset, followed by a fine-tuning step in which two loss functions are optimized: (1) a next-utterance classification loss, and (2) a language modeling loss. As a result, the model outperforms the existing systems by a significant margin obtaining 51% absolute improvement in perplexity on the validation dataset.

Given that many works successfully engage unsupervised learning in various manners, there still remains the question: how does unsupervised pre-training work? An answer is formulated by (Erhan et al., 2010) and a possible explanation is that the pre-training guides the learning towards basins of attraction of minima that support better generalization from the training dataset. Therefore, it acts like a regularizer for the supervised fine-tunning phase, when the parameters are restricted to a relatively small space. This assumption is reinforced by the results that show an effectiveness' upgrade of pre-training as the number of units per layer increases, a better generalization performance, but worse training errors, and worse performance than random initialization for small networks, all characteristics of regularization. They also show a growth in the probability of finding a local minima by increasing the depth of a network with random initialization, compared to an unsupervised pre-training.

The most important advantage of pre-training is the possibility of using unlabeled data, which is really helpful given the high costs of data annotation. Therefore, the effect of pre-training with very large datasets observed in the experiments is the most surprising result of the paper (Erhan et al., 2010): the early examples determine the basin of attraction for the remainder of the training and the

supervised fine-tuning cannot escape from it. The hypothesis is that those examples induce changes in the magnitude of the weights, which decreases the number of regions accessible to the stochastic gradient descent procedure. This is why, in a large-scale setting, the influence of unsupervised pre-training is still present, in contrast to the classical regularizers, when the effect disappears with more data.

Nevertheless, fine-tuning large pre-trained models is parameter inefficient, because each task requires an entirely new model. A compact and extensible model is needed in order to solve this shortcoming. For this purpose, (Houlsby et al., 2019) introduce adapter-based tuning, which achieves a mean GLUE score of 80.0 on several text classification tasks, compared to 80.4 achieved by full fine-tuning, using 1.3 task-specific parameters in total, compared to 9. This method also facilitates continual learning (training on a sequence of tasks) and multi-task learning (training on simultaneous tasks).

3 Model

The system used in this paper is a *semantic frames* system (Li et al., 2017). It represents the dialogue as a set of slot-value pairs and at each step t, given the user utterance u_t, the agent takes an action a_t, which can be either the final result or a request for a value of an empty slot. The architecture consists of two parts: a **User Simulator** module and a **Dialogue Manager** module.

The purpose of the **User Simulator** is to interact with the Dialogue Manager in order to train a policy for an agent. First, a user goal is chosen randomly from the goals' pool and is unknown for the agent, but it tries to help the user to accomplish it during the dialogue. The goal consists of two types of slots:

- *inform slots* - represent the constraints imposed by the user, hence their values are known (e.g. {movie_name: "deadpool", city: "Madison Heights", date: "saturday", number_of_people: "5"}).
- *request slots* - represent the values that the agent should provide, hence they enclose unknown values to the user (e.g. {price, start_time, critic_rating}).

Then, the user utterance u_t is generated following the **Agenda-Based** model (Li et al., 2016): the user has an internal state s_u composed of a goal G and an agenda A. The goal consists of constraints C and requests R. At each step t, the user simulator generates the user action $a_{u,t}$ based on the current state $s_{u,t}$ and the last agent action $a_{a,t}$ and updates the current state $s'_{u,t}$.

Natural Language Generation (NLG) is the module that generates natural language text for the user dialogue actions. For better results, a hybrid approach is used, including a model-based NLG and a template-based NLG. The model-based NLG is an LSTM decoder, which takes a dialogue action as input and generates a sentence with slot placeholders. If the sentence can be found in the predefined templates, the template-based NLG is applied for filling the slots, otherwise, the utterance generated by the model-based NLG is used.

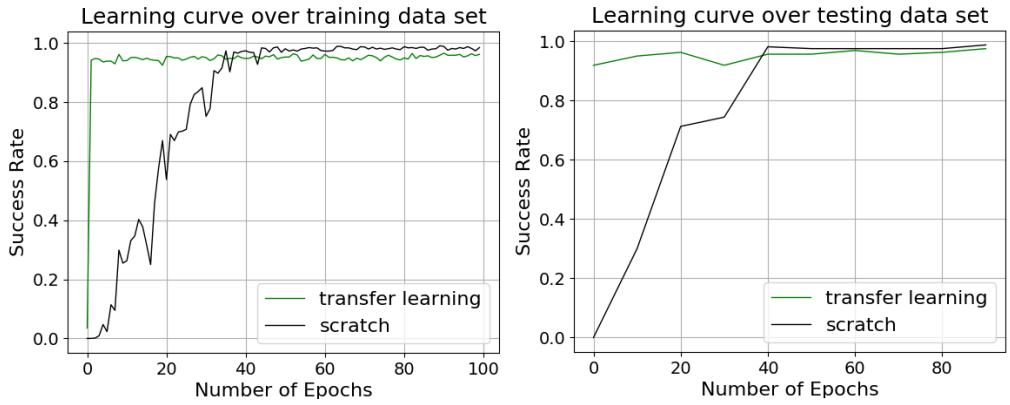
Natural Language Understanding (NLU) is the opposite to the NLG module: it takes as input an utterance and determines the user's *intent* and the set of *slots* associated with it (e.g. {movie_name: "deadpool", date: "saturday", number_of_people: "5"}), in order to form a semantic frame. It is implemented with an LSTM and its objective is to maximize the conditional probability of the slots and the intent, given the utterance.

The **Dialogue Management (DM)** includes two submodules: **Dialogue State Tracker** and **Policy Learning** module. The goal of the **Dialogue State Tracker** is to build a representation of the current state for policy learning, using the semantic frame received from the NLU component. It keeps the history of the user utterances, system actions and the query results from the Knowledge Base.

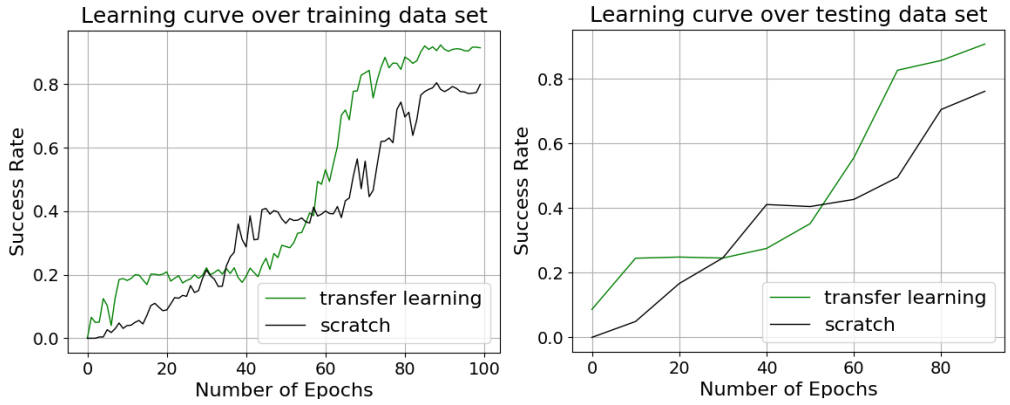
Policy learning module generates the next action of the system a_t according to the policy $\pi = P(a|s)$, given the current state s_t, in order to accomplish the user goal in the smallest number of steps. The state s_t includes the latest user action, the latest agent action, turn information, history dialogue turns and the available database results. A DQN (Mnih et al., 2015) is used to approximate the state-action function $Q(s, a|\Theta)$ and contains the experience replay mechanism.

4 Dataset

In order to study the impact of transfer learning in multiple domains, we choose MultiWOZ



(a) Hospital with pre-training on Attraction domain.



(b) Taxi with pre-training on Train domain.

Figure 1: Small number of extra slots in target domain.

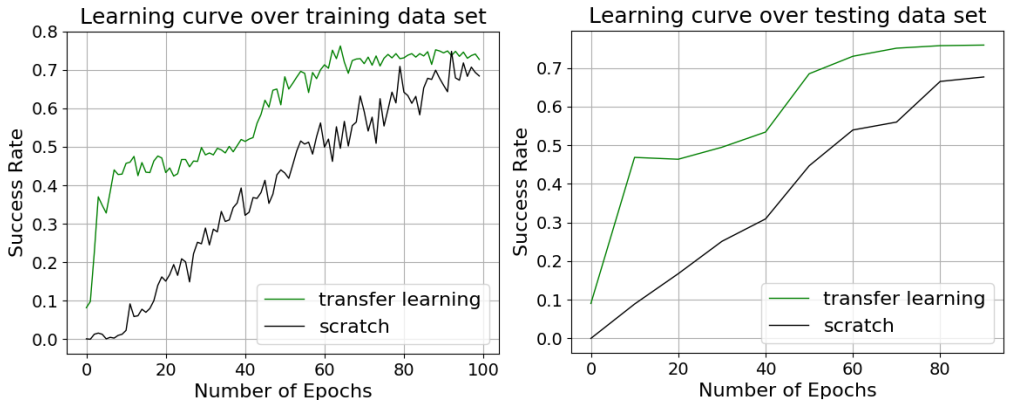
2.0 (Budzianowski et al., 2018), a large-scale multi-domain corpus of natural human-human conversations, collected through the Wizard-of-Oz framework (Kelley, 1984). It contains about 10000 samples from seven domains, with an average of turns per dialogue between 8.9 and 15, depending on the domain. From this dataset, we select the following five domains: **hotel**, **attraction**, **train**, **taxi**, **hospital**, and also keep **movie** and **tourist** domains used by Ilievski et al. (2018). These domains are grouped in source-target pairs according to their common slots, resulting five new opportunities for transfer learning. The total number of slots for source and target domains, respectively, as well as the amount of common slots is presented in Table1. We call *extra source/target slots* the difference between the total domain slots and the common ones.

The goals can be divided into two categories depending on whether they contain request slots or not. In the first case, the user sends a list of inform slots to the agent and the agent should accomplish the task respecting the constraints imposed by the user. In the second case, the user sends a list of request slots besides the list of inform slots, and the agent should accomplish the task and answer to the user's questions.

	Source Domain	Target Domain	Source Slots	Target Slots	Common Slots
1	movie	restaurant	6	9	3
2	restaurant	tourist	9	9	6
3	**hotel**	**attraction**	13	9	5
4	**train**	**taxi**	9	6	4
5	**movie**	**hotel**	17	13	5
6	**tourist**	**hotel**	9	13	6
7	**attraction**	**hospital**	9	4	3

Table 1: Number of slots per domain

This is a noteworthy detail, because it can influence the success rate through the experience accumulated in the warm start phase. In this phase, a fixed-size buffer is filled with experiences from positive-outcome conversations. Thus, the learning process gains a boost when having to self-calibrate based on an experience which will lead



(a) Attraction with pre-training on Hotel domain.

Figure 2: Medium number of extra slots in target domain.

to goal-achievement. We have noticed that the best results are obtained with warm start on no-request goals, following that the agent will manage to achieve request goals during training. As we increased the percentage of request goals in the warm start buffer, the overall success rate decreases and the learning curve becomes less smooth.

The total number of goals per domain is distributed as follows:

- 3000 training and 400 testing user goals in hotel, train and attraction;
- 2000 training and 200 testing user goals in taxi datasets;
- 80 training and 15 testing user goals in hospital dataset.

5 Experiments

The experiments are executed on overlapping domains, with the setup from (Ilievski et al., 2018) and running each experiment 10 times, with $n_{epochs} = 100$ epochs. The second set of experiments mimic testing on extension domain by restricting the slots in the source domain to the common ones. The warm start technique with experience replay buffer is used for both *transfer learning* and *scratch* agent (the same version, but without transfer learning), and the experience buffer is flushed when the agent reaches, for the first time, a success rate of 0.3. We also keep the maximal number of allowed turns per dialogue $n_{max\_turns} = 20$ in most experiments, except in the case of attraction domain with pre-training on hotel. In this situation, the number of turns is too small compared with the number of slots from the hotel domain, and the agent trained on the source domain is not able to learn. Consequently, we increased n_{max_turns} to 40 turns.

5.1 Different Domains

The first set of experiments aims to analyze the convergence for the agent with and without transfer learning on new domains. We group the experiments in three categories, according to the number of extra slots in the target domain, as follows: 1. small number of extra slots (less than 2 slots); 2. medium number of extra slots (between 3 and 6 slots); and 3. big number of extra slots (greater than 6 slots). We are interested in the improvement transfer learning brings to the success rate and the convergence pace.

Figure 1 presents the learning curve for the target domains with a small number of extra slots. For hospital domain with pre-training on attraction, both agents converge to a success rate greater than 95%, but the agent with transfer learning converges in a few epochs (<5), while the scratch agent needs 40 epochs to reach similar accuracy values. In the case of taxi domain with pre-training on train domain, the improvement of transfer learning is 15% for train dataset and 19% for test dataset. In absolute terms, the success rate increases from 80% to 91% for train dataset and from 76% to 91% on test dataset.

For the attraction domain with pre-training on hotel, presented in Figure 2, the model obtained with transfer learning has a success rate 7% higher than that of the scratch model on train dataset. This denotes an increase from 68% to 73% in absolute terms. For test dataset, transfer learning im-

proves the success rate from 68% to 76% or with 12% in relative terms.

The last category registers the lowest overall success rate for both agents and we consider that these results stem from the large number of extra slots in the target domain. The learning curves are illustrated in Figure 3 and the success rate is similar for train and test datasets. Hotel domain with pre-training on movie has a success rate of 27% with transfer learning and 8.5% without transfer learning, with an improvement of 217%. While the same domain with tourist as source domain of transfer learning, registers a relative boost of 737%, from 4.3% to 36%.

5.2 Same Domains, Different Number of Slots

The second set of experiments targets the evolution of the success rate according to the number of extra slots, both in the source and target domain. The selected experiment evaluates the hotel domain with pre-training on tourist dataset, given they each have large number of slots with few common ones (see Table1). We keep the setup for the other parameters and only change the number of extra slots in one domain, while the other remains constant.

	Source Slots	Target Slots	Scratch Score	TL Score	Scratch Epochs	TL Epochs
1	6	6	0.81	0.88	100	40
2	7	6	0.81	0.86	100	70
3	8	6	0.81	0.84	100	70
4	9	6	0.77	0.83	100	50
5	6	9	0.55	0.72	100	100
6	7	9	0.57	0.63	100	100
7	8	9	0.54	0.70	80	100
8	9	9	0.56	0.70	100	100

Table 2: Source Slots number influence

The final success rate on the test dataset for constant slots in target domain is summarized in Table 2. When the target contains only the common slots, we observe a decrease of the success rate with less than 2% with each extra slot added in the source domain, for the model trained with transfer learning. However, it is still by 6.6% greater than the success rate of the agent trained with any other prior knowledge. An interesting fact is that the same test with the common slots plus three extra slots in target domain has the effect of diminishing the success rate by an average of 15% compared with the previous situation. At the same time, the improvement over the scratch agent is equal to 24%.

	Source Slots	Target Slots	Scratch Score	TL Score	Scratch Epochs	TL Epochs
1	6	6	0.81	0.88	100	40
2	6	7	0.75	0.85	90	40
3	6	8	0.63	0.79	70	100
4	6	9	0.55	0.72	100	100
5	6	10	0.53	0.59	100	100
6	6	11	0.09	0.52	100	90
7	6	12	0.01	0.44	90	100
8	6	13	0.0	0.39	10	100
9	9	6	0.77	0.83	100	50
10	9	7	0.71	0.82	90	100
11	9	8	0.65	0.76	100	100
12	9	9	0.56	0.70	100	100
13	9	10	0.54	0.59	100	100
14	9	11	0.11	0.52	100	100
15	9	12	0.04	0.42	90	100
16	9	13	0.04	0.36	100	60

Table 3: Target Slots number influence

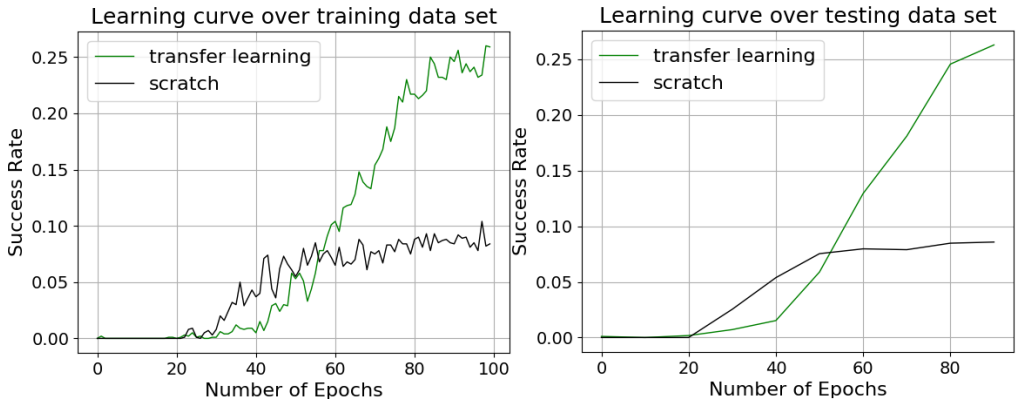
As expected, the number of extra slots in target domain has a bigger influence over the final results. Therefore, the relative average decrease of the success rate for the agent with transfer learning is 9.5% for each extra slot, while the source domain contains only common slots. Another three slots added to the source dataset generates an average decrease of 3.6%, relative to the previous test. In comparison with the scratch agent, the improvement increases from 79% in the first case, to 274% in the latter.

6 Conclusions

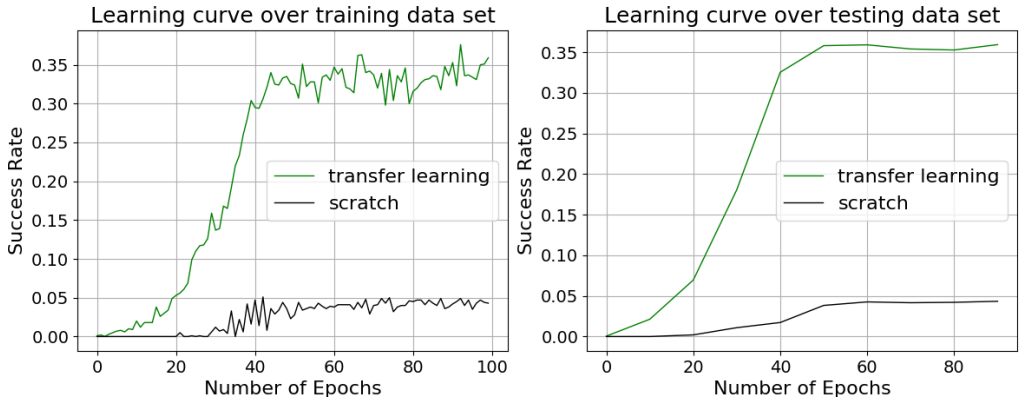
In this paper, we study the factors that influence the success of transfer learning approach in Reinforcement Learning-based Goal-Oriented Chatbots and demonstrate the results on five new cases of overlapping domains. We found that a big number of different slots between the source domain and the target domain leads to a smaller success rate. Even so, the transfer learning mechanism brings a betterment of over 79% over the agent trained with no prior knowledge.

The outcomes encourage the use of transfer learning with warm start on various cases of overlapping and extending source and target domains. However, the optimal selection in terms of hyperparameters of the system, such as the number of epochs or the number of maximum turns in a conversation, need to be determined for each particular scenario. They are, after all, directly influenced by the amount of data and its characteristics: number of slots, types and distribution of goals, and the degree of overlapping between source and target slots.

Further work involves developing wider experiment scenarios for hierarchical deep reinforce-



(a) Hotel with pre-training on Movie domain.



(b) Hotel with pre-training on Tourist domain.

Figure 3: Big number of extra slots in target domain.

ment learning system and introducing the transfer learning approach into this architecture when the sub-tasks share slots. Moreover, we can imagine other transfer learning setups such as sharing sub-tasks as the learnt common part, instead of slots, from one composite task to another. All these attempts have the objective of gaining more context information and better performance with less annotated data, which is onerous to obtain.

References

Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo 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, Brussels, Belgium, October 31 - November 4, 2018*. pages 5016–5026.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, and Pascal Vincent. 2010. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research* 11:625–660.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*. PMLR, Long Beach, California, USA, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. http://proceedings.mlr.press/v97/houlsby19a.html.

Vladimir Ilievski, Claudiu Musat, Andreea Hossmann, and Michael Baeriswyl. 2018. Goal-oriented chatbot dialog management bootstrapping with transfer learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*. pages 4115–4121.

John F. Kelley. 1984. An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems* 2:26–41. https://doi.org/10.1145/357417.357420.

Xiujun Li, Yun-Nung Chen, Lihong Li, and Jianfeng

Gao. 2017. End-to-end task-completion neural dialogue systems. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017 - Volume 1: Long Papers*. pages 733–743.

Xiujun Li, Zachary C. Lipton, Bhuwan Dhingra, Lihong Li, Jianfeng Gao, and Yun-Nung Chen. 2016. A user simulator for task-completion dialogues. *CoRR* abs/1612.05688.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu A., Joel Veness, and Marc G. Bellemare et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529–533.

Baolin Peng, Xiujun Li, Lihong Li, Jianfeng Gao, Asli Celikyilmaz, Sungjin Lee, and Kam-Fai Wong. 2017. Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. pages 2231–2240.

Iulian V. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*. pages 3776–3784.

Pei-Hao Su, Milica Gasic, Nikola Mrksic, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Continuously learning neural dialogue management. *CoRR* abs/1606.02689.

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*, Curran Associates, Inc., pages 3104–3112. http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf.

Ashish Vaswani, Noam Shazeer, Niki Parmar, and Jakob Uszkoreit et al. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. pages 6000–6010.

Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*. pages 438–449.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *CoRR* abs/1901.08149.