Curriculum Learning Based on Reward Sparseness for Deep Reinforcement Learning of Task Completion Dialogue Management

Atsushi Saito
Nextremer Co., Ltd., Tokyo, Japan.
atsushi.saito@nextremer.com

Abstract

Learning from sparse and delayed reward is a central issue in reinforcement learning. In this paper, to tackle reward sparseness problem of task oriented dialogue management, we propose a curriculum based approach on the number of slots of user goals. This curriculum makes it possible to learn dialogue management for sets of user goals with large number of slots. We also propose a dialogue policy based on progressive neural networks whose modules with parameters are appended with previous parameters fixed as the curriculum proceeds, and this policy improves performances over the one with single set of parameters.

1 Introduction

Learning in environments that give agents sparse and delayed reward is still a central research issue in reinforcement learning, while there are remarkable successes of deep reinforcement learning methods(Mnih et al., 2016; Bellemare et al., 2016; Ostrovski et al., 2017; Vezhnevets et al., 2017; Riedmiller et al., 2018).

The problem on sparse and delayed reward appears in reinforcement learning for task oriented dialogue agents. Contrary to single turn interactions such as chit-chat or question answering (Serban et al., 2016; Li et al., 2016), task oriented dialogue agents often are required to retrieve information from external knowledge bases and to learn the way how the agent reasons with progression of dialogue tasks over multiple dialog turns(Young et al., 2013; Williams et al., 2017). This long term process, however, makes it difficult for Markov Decision Process to identify the part of an action sequence that affects progress of dialogue tasks over multiple turns. Thus, typical agents must decide from a positive reward, which is obtained from successful task completion, only at the last turn.

It is inevitable for practical scalability to use sparse reward functions, because designing complicated and dense reward criteria over multiple turns involves domain knowledge and human annotators to evaluate dialogue history of large size. In particular, our aim is to train dialogue policy agents that cannot obtain positive rewards until the last turn.

While general and scalable frameworks of task completion dialogue management have been proposed recently, these frameworks still have had reward sparseness problem. Li et al. (2017) proposed a general neural dialogue framework which has scalability and features to solve information retrieval tasks (TC-Bot), which extended a previous work on information retrieval dialogue system (called KB-Info-Bot) to access external knowledge base (Dhingra et al., 2017). While they firstly proposed a robust end-to-end modularized neural dialogue system with separated and independently trainable modules, which are natural language understanding, dialogue management, and natural language generation, difficulty in reinforcement learning with sparse rewards still remains for their learning method with deep Q-networks (DQN).

In this paper, we propose curriculum learning based on reward sparseness of user goals, and agents using progressive neural networks (Rusu et al., 2016a) to improve the curriculum learning.

Our contribution is two-fold. First, our curriculum learning makes it possible to learn sets of user goals with large number of slots for which TC-Bot failed to learn. As the simulation epoch increases, the minimum number of slots that user goals contain increases.(See an overview in Figure 1) For example, the minimum number of slots is two for the first 200 simulation epochs, and is four for the next 200 ones, and agents are finally

Proceedings of the 2018 EMNLP Workshop SCAI: The 2nd Int'l Workshop on Search-Oriented Conversational AI, pages 46–51
Brussels, Belgium, October 31, 2018.
ISBN 978-1-948087-75-9

trained with user goals that contains at least 10 slots. In other words, the more simulation epoch proceeds, the more sparse reward is obtained from environments. There are two practical advantages of this curriculum: (1) our curriculum is domain free and (2) curriculum data preparation is easy because our curriculum only depends on the number of slots of user goals.

Second, the proposed application of progressive neural networks improves knowledge transfer from models trained for easier curriculum data to models trained for harder one. Progressive neural networks have multiple columns with weight parameters. At first a progressive neural network has single column to be trained, then another column is appended with new parameter set. All parameters of previous columns are frozen when appended column is training, and the appended columns can exploit information from frozen columns. Our aim is to apply this progressive freezing mechanism to exploit information of the parameters that are trained with *easier user goals* of our curriculum, when the latest appended column is in training with *harder user goals*. This progressive exploitation is expected to overcome the difficulty in the setting that agents start reinforcement learning with *the hardest user goals*.

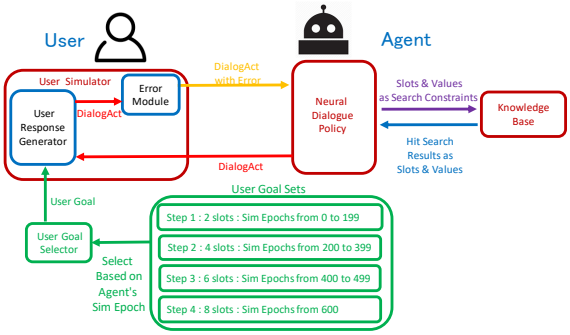


Figure 1: Overview of the way how to switch user goals.

2 Related Work

Task Oriented Dialogue One of the most popular models to learn task oriented (or goal oriented) dialogue is Partially Observable Markov Decision Process(POMDP) (Young et al., 2013; Verena Rieser,, 2010; Gasic et al., 2013). Another line of research is end-to-end neural modeling (Serban et al., 2016; Williams and Zweig, 2016; Liu and Lane, 2017a,b; Liu et al., 2018). While methods based on supervised learning are proposed in (Bordes et al., 2017; Wen et al., 2017), they come with the uncertainty of model performance for unknown data of interactions with humans. (Dhingra et al., 2017) proposed Reinforcement Learning dialog agent for learning the way how to access information of external knowledge base.

Progressive Neural Networks Originally, the notion of progressive neural networks is proposed in the research to transfer learning across multiple tasks and foreknowledge task similarity (Rusu et al., 2016a). Comparing with the original use of progressive neural networks, in our application, each column is *not necessary to be trained until convergence*, that is, our purpose is to provide the last column supplemental information, which is transfered from parameter weights obtained in environments with more dense reward. An application in robotic manipulations show, the way to adopt models that are trained in 3D simulation environments to real world physical environments (Rusu et al., 2016b). Similar to the approach of this paper, they tried to avoid designing complicated reward functions for application settings in real world.

Curriculum Learning The first proposition of the concept of curriculum learning is in (Bengio et al., 2009). While their curriculum data sets are based on complexity of shapes and graduation of colors to train image recognition models and the vocabulary size to train language models, our proposed curriculum data is based on the number of slots of user goals to solve goal oriented dialogue tasks and yields a kind of sub tasks that we can regard filling one slot as a sub task of filling two or more slots. The curriculum data set used in our experiment was created from slot types and their values of the movie search data set in (Li et al., 2017). The proposed method trains progressive neural networks to transfer knowledge across sets of user goals, and a theoretical relationship between transfer learning and curriculum learning is studied in (Weinshall et al., 2018).

3 Reinforcement Learning for Task Completion Dialogue Management

Task Completion Dialogue

Task completion dialogue management contains the following elements: user goals, task completed status, user simulators. These elements constitute reinforcement learning environments. More spe-

cific descriptions are as follows.

User goals: User goals contain two kinds of information : (1) pairs of slots and the values that users want to inform to systems as a constraint of items to be retrieved from knowledge base and (2) slots whose values are unknown for agents but they want to obtain the values of these slots.

Definition of task completed status: The dialogues between agents and users is defined as successful, if only if agents have proposed the slots and values based on retrieved information from knowledge base such that the following two conditions are satisfied: (1) these slots and values satisfy the constraints of user goals and (2) proposed slot types are requested in user goals.

User simulators: User simulators send a dialogue act, which provides a representation of the hidden semantics of a user utterance. There are two kinds of dialogue acts: (1) ones depending on slot types act like *inform* or *request* whose example is represented as a pair $(inform, \ movie \ name)$ and (2) ones independent of them such as *greeting* or *completing the task* etc.

Reinforcement Learning Agents and Environments

In here, we provide an explanation on actions of agents and state representations and reward functions which constitute Markov reward models of task completion dialogue management.

Agents' actions: Actions of reinforcement learning agents are dialogue acts and each dialogue act has at most one slot. The number of actions, which is also the dimensionality of action vectors, is the sum of the following: the number of inform slots, the number of request slots, and the number of actions that are independent on slot types.

State representations: State representations that agents can observe contain multiple kinds of vectors. These vectors include binary vectors representing subsets of inform slots or request slots, and include one-hot vectors representing current turn number. These vectors are necessary for agents to recognize progress of dialogue tasks. The state representation also contains information from external knowledge base such as lists of items in knowledge base that satisfy the users' requests and the sizes of these lists. State representations at time t also contain one-hot vectors of the agent's action at time $t-1$.

Reward functions: A large positive reward $2T_{max}$ is given to agents if dialogue status have been successful, and a negative reward $-2T_{max}$ is given for the failed status, where T_{max} is the maximum number of dialogue turns. We note that each of agent and user can send an utterance at most a half of T_{max} times. Additionally, for each turn, the negative reward -1 is given to the agents.

Finally, we describe the way to update deep Q-networks and a note on initialization of experience replay memory (ERM). Updates of Q-networks are performed with Bellman Equation and Mean Squared Error(MSE). An experience replay memory stores the transitions of agents. During initial experience, to avoid the cold start problem, agents use rule based policy, which essentially request each slot type, and stores the transitions obtained by this rule based policy. Then, agents start the training phase of deep Q-learning. Once DQN agent's performance on success rate overtakes rule based policy, ERM is set to an empty list.

4 Proposed Methods

Curriculum of User Goals

In here, we describe our curriculum of user goals. The purpose of this curriculum is to investigate the performance of dialogue management for the set in which *only user goals with large number of slots* are contained. Four sets of user goals were prepared. The minimum number of slots for each user goal set is showed in Table 1.

Set	inf	req	all
A	1	1	2
B	2	2	4
C	3	3	6
D	6	2	8

Table 1: The minimum number of slots for each user goal set. The three labels **inf, req, all** respectively correspond to the number of inform slots, request slots, and all slots.

Step	Range	User Goal Set
1	0-199	$A \cup B \cup C \cup D$
2	200-399	$B \cup C \cup D$
3	400-599	$C \cup D$
4	600-1200	D

Table 2: Set of user goals selected in each simulation epoch range of our curriculum. The sets A,B,C,and D are defined in Table 1

The pairs of the ranges of simulation epochs and the corresponding set of user goals are showed in

	slot type	value
inform slots	city	Seattle
request slots	theater	Unknown

Table 3: An example of easy user goals in Step 1. Users can obtain one of many names of theaters in Seattle.

	slot type	value
inform slots	starttime	19:00
	genre	history
	date	August 31
	actor	Tom Hanks
	city	Seattle
request slots	theater	Unknown
	moviename	Unknown

Table 4: An example of hard user goals in Step 4. Users can obtain a name of movie **The post** and a name of theater **Admiral Theater** for *moviename* slot and *theater* slot, respectively.

Table 2. We consider the difficulty of sets of user goals as follows: the less slots a set of user goals contains, the easier the set of user goals is. In our curriculum, at first, a user goal is randomly sampled from the union of all sets of user goals defined in Table 1. Then, as the step proceeds to the next one, the easiest set of user goals is removed from the union.Thus, at last, the set D which contains only hardest user goals is used for simulation. Examples of user goals are showed in Table 3 and Table 4.

We note that our curriculum training method takes into account the possibly varying order of slots during training, because agents must fill all slots which users have informed with an arbitrary order in our experiments.

There are two remarks of our curriculum. First, the longest dialog episodes with no redundant agent's action are yielded from the set D. That is, the proposed training process does not concatenate dialogs from different sets as training goes on. Second, each of four sets of user goals in Table 1 has a variety of types of slots, because each set of user goals was created by choosing random pairs of slot type and its value.

Thus, the proposed curriculum can be created from all kinds of data sets of user goals for task completion dialogue based on slot filling, and the proposed training process does not depend on data sets of user goals.

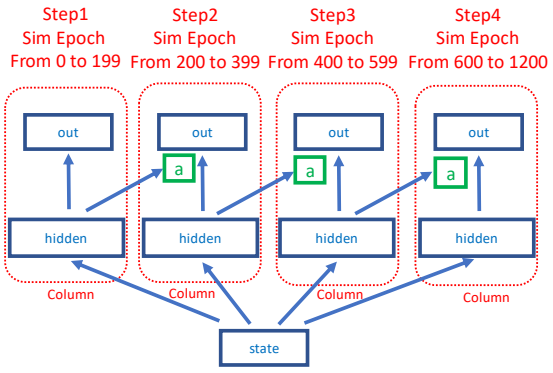


Figure 2: Training Process via Progressive Nets: As the curriculum step increases, new columns are appended. Green modules represent adaptors.

Progressive Neural Dialogue Policy

In here, we describe the notion of progressive neural networks and its applications to our setting (See Figure 2 for an overview). We define feature vectors of frozen columns as $h_{i-1}^{(<k)} = [h_{i-1}^{(1)}; h_{i-1}^{(2)}; ...; h_{i-1}^{(k-1)}]$ of dimensionality $n_{i-1}^{(<k)}$, where the symbol ; denotes concatenating. Progressive networks have lateral connections through which we leverage prior knowledge to previously learned features and they have their own activation functions. Before feeding the lateral activations into linear layer, we multiply them by a trainable scalar called scaling factor, initialized by a random small value to adopt for the different scales of the different inputs. The hidden layer of the non-linear adapter is a projection onto an n_i dimensional subspace. We denote $W_i^{(k)} \in \mathbb{R}^{n_i \times n_{i-1}}$ as the weight matrix of layer i of column k, and denote $U^{(k:j)} \in \mathbb{R}^{n_i \times n_{i-1}}$ as the lateral connections from layer $i-1$ of column j, to layer i of column k and h_0 is the network input. Thus, the output of the i-th layer of k-th column is:

$$h_i^{(k)} = \sigma(W_i^{(k)} h_{i-1}^{(k)} + U_i^{(k:j)} \sigma(V_i^{(k:j)} \alpha_{i-1}^{(<k)} h_{i-1}^{(k)}))$$

, where $V_i^{(k:j)}$ is the projection matrix and $\alpha_{i-1}^{(<k)}$ is the scaling factor, σ is ReLU function, and bias terms are omitted. In our curriculum learning, an agent has a deep Q-network represented as a progressive neural network, and new column is appended when the step in Table 2 is count up. In our settings, the number of hidden layers is one, and its size of units is 80.

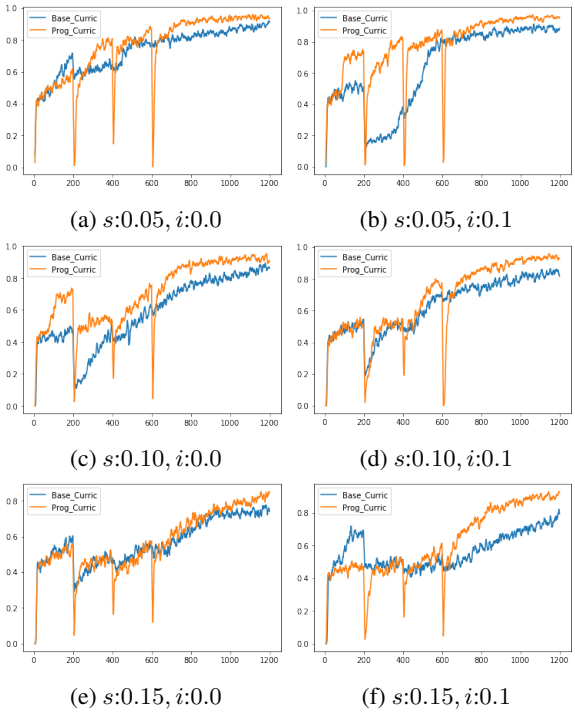
5 Experiments

Reinforcement Learning Environments and Data Set The curriculum data of user goals for the experiments was created from the movie searching data set used in (Li et al., 2017). The same reinforcement learning environment and user simulator in (Li et al., 2017) was used for the experiments.

User Simulator In our experiments, user simulators try to let dialogue agents fill slots which users have informed. The simulators also inform values of slots which users have requested as constraints to retrieve values from a data base. If the simulators have a slot type which they have not informed yet, they also inform its value. The simulators inform the value *I don't care* if agents have requested values of a slot type which is not contained in inform slots of user goals. For example, the simulators with the user goal showed in Table 4 send the message *I don't care*, when agents have requested the value of slot type *price*, because *price* slot type is not contained in inform slots in Table 4.

Setup of Experiments RMSprop was used as the optimizer. The hyper parameters of the optimizer were set to the following values: the learning rate, the decay rate and the momentum were, 0.001, 0.999, and 0.1, respectively. With the way similar to (Li et al., 2017), the error control model that has two kinds of errors: slot level and intent level was used. In the experiment, slot level and intent level correspond to the case where the slot name is correctly recognized but the slot value is wrong and the case where a dialogue act itself is wrongfully recognized, respectively. For each simulation epoch there are 100 episodes of dialogue between users and agents. In each episode of dialogue, a user can send an utterance at most a half of T_{max} times and an agent can perform in the same way.

Results The success rate (moving average with window of size 7) of each simulation epoch is shown in Figure 3. This figure shows that progressive neural network can make learning faster, noting that agents were trained for the set D in Table 1 for simulation epoch more than 600. The simulation in which agents are trained with only the user goal set D for all simulation epochs was also executed. The success rates for this hardest simulation were 0.0 for all simulation epochs and for all of six error settings (omitted in Figure 3). In particular, for the success rates in 3b and 3c, progressive neural networks improve the performance for all simulation epoch ranges in Table 2, while the success rates for progressive networks drop at switching epochs(200, 400, 600).



(a) s:0.05, i:0.0 (b) s:0.05, i:0.1

(c) s:0.10, i:0.0 (d) s:0.10, i:0.1

(e) s:0.15, i:0.0 (f) s:0.15, i:0.1

Figure 3: For slot level error ϵ and intent level error δ, the caption of each figure is written as s:ϵ,i:δ. Blue lines correspond to curriculum learning deep-Q-networks, and Orange lines correspond to progressive neural network models.

6 Conclusion

In this paper, we introduce a curriculum on reward sparseness of user goals to tackle reward sparseness problem in reinforcement learning for task completion dialogue management, and this curriculum makes it possible to learn via reinforcement learning of dialogue management task using user goals with large number of slots. We also propose a method based on progressive neural networks to improve learning performance. Experiments show that progressive neural networks enhance the curriculum reinforcement learning.

Acknowledgements

The author wishes to thank Taichi Iki, Yuki Sekizawa, and the anonymous reviewers for helpful comments.

References

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. 2016. Unifying count-based exploration and intrinsic motivation. In *NIPS2016*.

Yoshua Bengio, Jerome Louradour, Ronan Collober, and Jason Weston. 2009. Curriculum learning. In *ICML2009*.

Antoine Bordes, Y-Lan Boureau, and Jason Weston. 2017. Learning end-to-end goal-oriented dialog. In *ICLR2017*.

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 *ACL2017*. pages 484 – 495.

Milica Gasic, Catherine Breslin, Matthew Henderson, Dongho Kim, Martin Szummer, Blaise Thomson, Pirros Tsiakoulis, and Steve Young. 2013. Online policy optimisation of bayesian spoken dialogue systems via human interaction. In *IEEE ICASSP*. pages 8367–8371.

Jiwei Li, Michel Galley, Chris Brockett, Georgios P Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. In *ACL2016*.

Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. 2017. End-to-end task-completion neural dialogue systems. In *IJCNLP2017*.

Bing Liu and Ian Lane. 2017a. An end-to-end trainable neural network model with belief tracking for task-oriented dialog. *arXiv:1708.05956* .

Bing Liu and Ian Lane. 2017b. Iterative policy learning in end-to-end trainable task-oriented neural dialog models. In *IEEE Workshop on Automatic Speech Recognition and Understanding(ASRU)*.

Bing Liu, Gokhan Tur, Dilek Hakkani-Tur, Pararth Shah, and Larry Heck. 2018. Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems. In *NAACL2018*.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2016. Human-level control through deep reinforcement learning. *Nature* 518:529–533.

Georg Ostrovski, Marc G. Bellemare, Aaron van den Oord, and Remi Munos. 2017. Count-based exploration with neural density models. In *ICML2017*.

Martin Riedmiller, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degrave, Tom Van de Wiele, Volodymyr Mnih, Nicolas Heess, and Jost Tobias Springenberg. 2018. Learning by playing - solving sparse reward tasks from scratch. *arXiv:1802.10567* .

Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016a. Progressive neural networks. *arXiv:1606.04671* .

Andrei A. Rusu, Mel Vecerik, Thomas Rothrl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell. 2016b. Sim-to-real robot learning from pixels with progressive nets. *arXiv:1610.04286* .

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 *AAAI2016*.

Oliver Lemon Verena Rieser,. 2010. *Reinforcement Learning for Adaptive Dialogue Systems*. Springer. In chapter 2, section 2.1, page 10.

Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. 2017. Feudal networks for hierarchical reinforcement learning. *arXiv:1703.01161* .

Daphna Weinshall, Gad Cohen, and Dan Amir. 2018. Curriculum learning by transfer learning: Theory and experiments with deep networks. *arXiv:1802.03796* .

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 *EACL2017*.

Jason D Williams, Kavosh Asadi, and Geoffrey Zweig. 2017. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *ACL2017*.

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

Steve Young, Milica Gasic, Blaise Thomson, and Jason D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *IEEE* 101(5):1160 – 1179.