Noname manuscript No.
(will be inserted by the editor)

A Hierarchical Approach for Efficient Multi-Intent Dialogue Policy Learning

Tulika Saha · Dhawal Gupta · Sriparna Saha · Pushpak Bhattacharyya

Received: date / Accepted: date

Abstract This paper proposes a hierarchical method for learning an efficient Dialogue Management (DM) strategy for task-oriented conversations serving multiple intents of a domain. Deep Reinforcement Learning (DRL) networks specializing in individual intents communicate with each other, having the capability of sharing overlapping information across intents. The sharing of information across state space and the presence of global slot tracker prohibits the agent to reask known information. Thus, the system is able to handle sub-dialogues based on subsets of intents covered by different Reinforcement Learning (RL) models, thereby, completing the dialogue without again asking already provided information common across intents. The developed system has been demonstrated for "Air Travel" domain. The experimental results indicate that the developed system is efficient, scalable and can serve multiple intents based dialogues adequately. The proposed system when applied to 5-intent dialogue systems attains an improvement of 41% in terms of dialogue length as compared to a single-intent based system serving the same 5-intents.

Keywords Multi-Intent · Dialogue Management · Hierarchical · Deep Reinforcement Learning (DRL)

T. Saha
Indian Institute of Technology Patna, India
E-mail: sahatulika15@gmail.com

D. Gupta
Indian Institute of Technology Patna, India
E-mail: dhawal.gupta.iitp@gmail.com

S. Saha
Indian Institute of Technology Patna, India
E-mail: sriparna.saha@gmail.com

P. Bhattacharyya
Indian Institute of Technology Patna, India

1 Introduction

Dialogue systems are essentially described as chat bots wherein the humans communicate with a Virtual Agent (VA) through text or speech in order to attain a particular goal. In such a communication process, natural language plays a vital role [10]. In any goal-oriented dialogue systems, the dialogue manager plays the key role of deciding an action to be taken given any particular point in the conversation [19]. Dialogue being a temporal process can result in having innumerable dialogue states. This is often dealt with manually designed small space of constricted dialogue states and then the DM component is designed by hand with a tremendous amount of human effort and expertise. This makes the DM component rule-based [18]. The rapid advancement of Reinforcement Learning (RL) [13], more specifically Deep RL (DRL) [1], [21] algorithms have led to their extensive usage in various fields including DM that aims to learn the features as well as policies, jointly. This allows for the incorporation of larger and complex state space of dialogues, thus opening the possibility of creating more comprehensive dialogue managers that can handle numerous dialogue scenarios with ease, higher accuracy and consistency.

Numerous amount of works that are done in the past recommend considering DM as an optimization problem [15]. Other prominent works in the context of DM that use DRL are [5], [2], [6], [14], [27] etc. But such works lack diversity, i.e., those works are more related to the context of serving a particular dialogue scenario or intent of the user. More precisely, they focus on learning an optimal dialogue strategy that is meant to fulfill a particular goal or task of the user in a dialogue conversation. But in real world applications, the user generally wants to accomplish tasks which include getting several intents fulfilled in a single dialogue conversation with minimal effort and dialogue turns. Conversations in real time systems often do not work on single intent and vary amongst number of intents subject to a particular domain. Hence, in the current scenario of automation, there is a need of a multi-intent dialogue system which can converse with users in natural and rational manner. This requires the VA to optimally solve the user query based on different intents and dialogue states and continue until the user is satisfied. Apart from using RL in DM tasks, RL has also found its increased usage in Financial Markets. Authors of [20] made a thorough survey on how and why RL can be helpful in stock/forex prediction or trading. Apart from unrealistic assumptions these algorithms have taken, it highlights the need of automation of such tasks in a global scale and the demand for future research in these areas.

This paper presents a DRL based DM strategy in a multi-intent structure. The idea is to develop a system that is meant to be scalable across varied domains with minimal changes subject to different intents and slots on which the domain operates. Thus, the task of the VA is to learn a policy that tends to serve various intents of the user in a multi-intent framework of a particular domain in a dialogue conversation. This requires the VA to be intelligent, utilize the global information of the system efficiently, transfer overlapping information from one intent to another and serve the user with minimal number

of dialogue turns and accuracy. Hence, we develop a Markov Decision Process (MDP) for a multi-intent framework that finds its applicability across any domain. The multi-intent framework is primarily a hierarchical model where the top hierarchy decides which intent to serve in a supervised manner and the bottom hierarchy decides the action to be picked up in order to communicate with the user to achieve a sub-goal in an automated RL based approach. Q-learning [28], which is a model-free reinforcement learning algorithm has been employed in its DRL variant. The Double Deep Q-network with Prioritized Experience Replay (DDQN-PER) [26] has been used to learn a policy. The developed system is demonstrated for the "Air Travel" domain. Experimental results as compared to several baselines establish the efficacy of the proposed methodology.

The key contributions of this paper are the following :

- This paper presents a hierarchical approach to develop a Dialogue Manager which aims to serve and accomplish multiple intents of the user in a single dialogue conversation.
- Normally, system based on single intent, when applied to a real-life scenario of serving multiple intents, tends to take a lot of turns to make a valid database query because of multiple overlapping slots across various intents. The system proposed aims at reducing the number of turns while serving multiple intents by sharing the relevant and redundant information across state space of intents.
- A combination of different MDP and flow control across different intents, which streamlines the conversation, makes the process less arduous and more effective.
- The mechanisms of state space update and global slot tracker make the DM even more robust and scalable to real-life scenarios.

2 Related works

This section provides a brief description of the works done so far on RL based DM Strategy.

Authors of [22] proposed a Hierarchical Reinforcement Learning (HRL) approach based on *options* framework to learn policies in different domains for a single intent. Similarly, in [3], authors employed options based approach to treat individual domain as master domain whereas its individual intents as sub-domains to create a VA for multi-domain dialogue system. In [7], authors developed a DM strategy for multi-domain dialogue systems and applied it to the domains of hotels and restaurants for a single intent. However, such framework does not scale to modeling complex conversations by restraining their performance as domains and intents often share subtasks and slot space, respectively not defined in their approach. In [29], authors propose a divide and conquer approach for efficient policy learning where a complex goal-oriented task is broken into simpler subgoals in an unsupervised manner and then

these subgoals are used to learn a multi-level policy using HRL. Feudal Reinforcement Learning has been used with DQN in the work of [4] for learning policies in large domains; however, this particular work uses handcrafted feature functions to model policies. These works however, focus on proposing DM methodologies to handle multi-domain conversations with a single subtask/intent per domain. Whereas our work focuses on handling composite and complex, multi-intent dialogue conversations.

Apart from them, there are other significant works that aim to propose methodologies to learn DM policies for a single intent pertaining to a domain. In [5], authors developed an easy and open-sourced dialogue system using DRL for the restaurant domain and so the system evades from using hand-crafted features for learning an action-selection strategy without the use of the Natural Language Understanding (NLU) module. One of the limitations of this work is that even if the VA learns an optimal policy, its usability is restricted because of its dependence on the vocabulary. The system falters in out of vocabulary words and hence is difficult to be scalable in complex scenarios. In [7], authors developed a DM strategy for multi-domain dialogue systems and applied it to the domains of hotels and restaurants. For switching amongst different domains, they used a domain classifier. They claimed that their proposed method showed better scalability and was efficient in optimizing the performance of the system. However, slots of different domains and intents are not entirely discrete and are dependent on each other. Thus, lack of information sharing amongst the network can not be extended to real life scenarios. In [8], authors proposed a fast DRL approach that uses a network of DQN agents that skips weight updates during exploitation of actions. In [33], authors proposed an end-to-end RL approach to dialogue policy learning for a task-oriented scenario of guessing the famous person a user thinks of. However, the agent uses a sequence of Yes/No questions to retrieve the correct answer. Usage of Yes/No question makes dialogue conversation long and redundant along with being restricted to the agent database. Also this limits the dialogue to the case where agents head the dialogue by under utilizing the available communication bandwidth that the user can provide. Similarly, [16] also presented an end-to-end RL framework to overcome such issues and induce flexibility in dialogue conversations. In [9], authors presented an effective dialog policy learning to recover from automatic speech recognition and natural language understanding errors making the policy robust against noisy environments. In [12], authors proposed a Transfer Learning based DM for a single intent per domain. However, their approach couldn't establish how these Transfer Learning approaches can be leveraged to develop a multi-domain based system. In [17], authors proposed a variant of DQN where the VA explores via Thompson sampling, drawing Monte Carlo samples from a Bayes-by-Backprop neural networks. In [2], authors proposed an end-to-end goal oriented dialogue based on memory networks to conduct proper conversations and issue API calls. In another such work [31], authors developed a network based end-to-end trainable task oriented system without the use of RL techniques. However, scalability of such approaches remains an open question as training of these

models requires huge amount of dialogue conversation data of any domain. Acquiring considerable amounts of conversational data itself poses a bigger issue. Apart from these, works such as [32] aim at identifying multiple user goals in a single user utterance, i.e., multi-intent classification. Their approach is based on exploiting n-gram features to identify intents and using segments of the sentence instead of the entire sentence to assign class labels.

2.1 Motivation

From the literature, it is evident that several works done earlier in the context of dialogues had shortcomings. Recent research focuses on merging the NLU and DM into a single module eliminating the need of NLU modules and creating a single model in order to avoid NLU fault chances. These types of models restrict the usability of trained policy only to situations where dialogue vocabulary matches the training corpus, change in vocabulary requires a new model to be trained from scratch which becomes cumbersome for continually evolving and online systems. Also works, which employed Deep RL techniques for the problem, incorporate vocabulary of the system as state representation without the use of NLU module. So, even if the VA learns an optimal policy, its usability is restricted because of its dependence on the vocabulary and hence is not scalable. Other approaches proposed require extensive dialogue data, demand huge computational cost for training such complex networks. Often scalability, reusability and reproducibility of these proposed models are not achievable in real life implementation scenarios. Also, majority of these works focus on serving single intent or task of the user in a dialogue conversation which is highly not desirable in practical scenarios.

Motivated by the inadequacy of the existing systems and approaches, this paper presents an approach to serve multiple intents of the user in a single dialogue conversation without discretizing information across intents using a hierarchical approach.

3 Problem Statement

The main objective of this particular task is to learn an optimal policy π^* for multi-intent task-oriented dialogue conversation between the VA and the user. A policy π is defined as $\pi(s) \rightarrow a$ which is a mapping from states to actions that typically depicts the behavior of the VA. π^* represents an optimal policy which maximizes the cumulative reward at the end of an episode. One episode of such an interaction is a series of states, actions and rewards:

$$s_0 \underset{a_0}{\longrightarrow} (s_1, r_0) \underset{a_1}{\longrightarrow} (s_2, r_1) \underset{a_2}{\longrightarrow} (s_3, r_2), ..., s_{n-1} \underset{a_{n-1}}{\longrightarrow} (s_n, r_{n-1})$$

where s_n indicates the state at time-step n, a_n the action and r_n the reward after the execution of the action a_i leading to the transition into the state s_{n+1}. Thus, the goal of the VA is to select actions in a way so as to maximize its

discounted future reward. The VA picks up optimal actions at every time-step based on the policy learnt as

$$a = argmax_{a \in A} \pi(s, i; \theta) \tag{1}$$

where A is the set of all available VA actions present in the system. i is the particular intent in control and θ represents all the parameters of the function approximator of the RL model (which is a Deep Neural Network in our case). The RL model takes as input the user utterance in the form of a state s which is obtained by several processings of the Natural Language Understanding module. It outputs an action a from the policy π which is presented to the user through a Natural Language Generation (NLG) framework to curate an end-to-end task-oriented system. However, learning an optimal dialogue policy for multi-intent conversations forms the core of this paper. In the context of the current work, *Intent* captures the main communicative intention of the user utterances. So, similarly *multi-intent* captures more than one intentions present in an user utterance. *Slots* are basically defined as the important information that are present in the user utterances. An example of an user utterance with its intent and valid slots is shown in Table 1.

Table 1: Example utterance with its intent(s) and valid slots

Utterance	show	me	flights	and	cheapest	fare	from	Pittsburgh	to	Denver
Slot	O	O	O	O	O	O	O	B-fromloc.-city_name	O	B-toloc.-city_name
Intent(s)	flight + airfare									

4 Proposed Methodology

In the current study, we aim to develop a multi-intent based DM strategy. This section presents the MDP for a single intent (applicable to all other intents of the system) followed by the description of the system to combine multiple intents in the VA, followed by the experiments and a short description of the NLU module consisting of the SF and IC.

4.1 Proposed MDP

A generic architecture of MDP is used. It finds its applicability in any domain having n number of intents and m number of slots per intent. The task of the VA is to elicit necessary information in the form of slots from the user based on the intent identified to make a valid database query so as to provide necessary and apt information based on the data elicited. This process continues until the user's task(s) is completed.

Table 2: Slots be elicited for each intent

Intent	**Slots Present**				
flight	Departure City (deptcity)	Arrival City (arrCity)	Depart Time (deptTime)	Depart Date (deptDay)	Class of the flight (class)
airfare	Departure City (deptcity)	Arrival City (arrCity)	Round Trip (RTrip)	Depart Date (deptDay)	Class of the flight (class)
airline	Departure City (deptcity)	Arrival City (arrCity)	Class of the flight (class)	-	-
ground service	City (GCity)	Transport Type (Tran-Type)	-	-	-
ground fare	City (GCity)	Transport Type (Tran-Type)	-	-	-

Table 3: Details of ATIS dataset

	ATIS	
	Train	**Test**
# of Utterances	4978	893
# of Tokens	61568	10090
# of Labels	127	

The proposed method is illustrated in the Air Travel domain. The approach is applied on Air Travel Information System (ATIS) [24] dataset where the user can have multiple intents. Therefore, the intents taken into account (from the ATIS dataset itself) to demonstrate the current work are as follows: "flight", "airfare", "airline", "ground service", "ground fare". The statistics of the dataset is shown in Table 3.

State Space : At any given time, the state space consists of a pair of states (S_1, S_2) which are described as follows:

• $\mathbf{S_1}$: It is a state which is a tuple of n variables corresponding to the number of intents in a particular domain. It keeps track of the current intent (output from the IC module) being serviced by the VA in the form of one hot encoded vector. This in turn invokes the appropriate RL model corresponding to the intent being served. State S_1 represents the top-level hierarchy of the DM. In the current context, the state space is as follows :

[**flight airfare airline ground-service ground-fare**]

Therefore, if the intent identified is "flight", the state space becomes [1 0 0 0 0] and so on.

• $\mathbf{S_2}$: For each of the intents, S_2 is a tuple of variables of the length equal to the number of slots required to be filled up for that particular intent. The necessary slots to be filled up for each of the intents individually taken into account for this particular task are described in Table 2. State S_2 represents the bottom-level hierarchy of the DM. So, for example, for the "flight" intent, S_2 is a tuple of five variables as shown below :

[**deptCity arrCity depDay deptTime class**]

These slot variables in turn correspond to confidence scores of different slots which are the probability values outputted from the SF module representing the confidence of the module in predicting different slot labels.

Precisely, state S_1 keeps track of the intents being served. State S_2 is the corresponding tuple of slot variables for that particular intent being served. The DRL model actions incite changes in S_2 only whereas the S_1 is altered by the IC module.

Table 4: Action Set

(a)

Type	ASK							
Action	askdeptCity	askarrCity	askdeptTime	askdepDay	askclass	askRTrip	askGcity	askTranType
Slots Filled	deptCity	arrCity	deptTime	depDay	class	RTrip	Gcity	TranType
Intent	flight, airfare, airline	flight, airfare, airline	flight	flight, airfare	flight, airfare, airline	airfare	ground service, ground fare	ground service, ground fare

(b)

Type	ASK				SALUTATION
Action	askDeptArr	askDateTime	askDateRtrip	askGcityTrantype	closing_conversation
Slots Filled	deptCity, arrCity	depDay, deptTime	depDay, Rtrip	Gcity, Trantype	-
Intent	flight, airfare, airline	flight, airfare	airfare	ground service, ground fare	All

(c)

Type	REASK/CONFIRM							
Action	reaskdeptCity	reaskarrCity	reaskdeptTime	reaskdepDay	reaskclass	reaskRTrip	reaskGcity	reaskTranType
Slots Filled	deptCity	arrCity	deptTime	depDay	class	RTrip	Gcity	TranType
Intent	flight, airfare, airline	flight, airfare, airline	flight	flight, airfare	flight, airfare, airline	airfare	ground service, ground fare	ground service, ground fare

Action Space : The action space constitutes of 21 unique actions categorized in three different classes, i.e., *Ask*, *Reask/Confirm* and *Salutation* such as "*askClass*", "*reaskGCity*" and so on shown in Table 4. Apart from having actions to fill individual slots, there are hybrid actions to fill two slots at the same time such as "*askdeptarrCity*", "*askGCityTtype*". Reask/Confirm actions act as tools to fill up any capability lacked by the SF module in terms of the confidence in understanding the information given by the user as slots tend to present a more natural conversational experience.

Reward Model : The reward model is designed in a way such that the immediate credits assigned to the VA in different instances of the dialogue are attributed differently so as to make the VA understand what it needs to do distinctively at different time-steps of the conversation. The reward model operates only on state S_2 of the state space. It is described as:

– *Case 1 :* The reward function at any other time-step except at the terminating or closing step is as follows :

$$R(s, a, s') = (w_1 * (\parallel \overrightarrow{NS} \parallel_1 \; - \; \parallel \overrightarrow{CS} \parallel_1)) - (w_2) \tag{2}$$

where $\overrightarrow{NS}$ represents the state vector for the new state s'. $\| \overrightarrow{NS} \|_1$ is the summation of the confidence scores of all the state variables in the state vector which is obtained after taking an action a in state s. $\overrightarrow{CS}$ represents the state vector for the current state s. $\| \overrightarrow{CS} \|_1$ is the summation of the confidence scores of all the state variables in the state vector for state s. w_1 is the weight over the difference of the summation of the two state vectors in state s and $s\prime$. w_1 is used to encourage the agent to act in a way so as to increase its confidence on the acquired slots. w_2 is used to encourage useful communication and discourage unnecessary iterations. Here, $w_1 = 8$ and $w_2 = 1$ for our experiments. All specific values were assigned through empirical analysis by conducting the parameter sensitivity tests, details of which are shown in Table 5. The table shows the results and analysis of the parameter sensitivity test conducted to assign values to w_1 and w_2. Multiple experiments were conducted by varying the values and combinations of w_1 and w_2. It was observed that the MDPs converged better and faster for the ideal case (chosen weights) and was then used for the remainder of the experiments.

- *Case 2 :* The reward function at the terminating time-step is subject to a checking condition (mentioned below). If the checking condition is satisfied, the agent gets the reward as follows:

$$R(s, a, s') = V * w_1 * \| \overrightarrow{CS} \|_1 \tag{3}$$

  Where $V \in \{0, 1\}$, obtained from the checking condition.
- *Case 3 :* If the checking condition is not satisfied, the reward function is :

$$R(s, a, s') = -w_1 * (\| \overrightarrow{EV} \|_1 - \| \overrightarrow{CS} \|_1) \tag{4}$$

  where $\overrightarrow{EV}$ is the state vector for the expected value. $\| \overrightarrow{EV} \|_1$ is the summation of the maximum expected confidence scores of different slots that adds up to be equal to n for MDP with n slots (the maximum expected confidence score for each slot being 1). This is done to penalize the VA for not filling the slots with higher confidence in the sense to rule out possibilities of mis-identification of any slot value.

The checking criteria is as follows : if the confidence scores of all the individual slots for a particular state S_2 i.e., if $\forall i, S_{2_i} \geq threshold$ (set to 0.7), then $V = 1$, otherwise 0 (which implies that the checking condition is not fulfilled).

The reward model proposed is motivated to train the agent to pick action from any context in the dialogue. The reward function proposed is general, having reusable structure for different tasks and domains with minimal changes.

4.2 Network of Deep Reinforcement Agents

We aim to optimize our multi-intent dialogue system using a network of Deep Reinforcement Learners/Agents, i.e., a network of DDQN-PER. Therefore,

Table 5: Results and analysis of the parameter sensitivity test

Trend	w_1	w_2	Observation
Ideal	8	1	All slots of a MDP was filled with confidence greater than a threshold
Increase w_1	9	1	All slots does not have confidence greater than a threshold
Decrease w_1	7	1	All slots were not filled and the confidence was less than a threshold
Increase w_2	8	2	All slots does not have confidence greater than a threshold and the VA didn't learn to pick up hybrid actions to fill slots
Increase w_2	8	3	All slots does not have confidence greater than a threshold and the VA didn't learn to pick up hybrid actions to fill slots
Increase w_1, w_2	9	2	All slots does not have confidence greater than a threshold
Decrease w_1, Increase w_2	7	2	All slots does not have confidence greater than a threshold

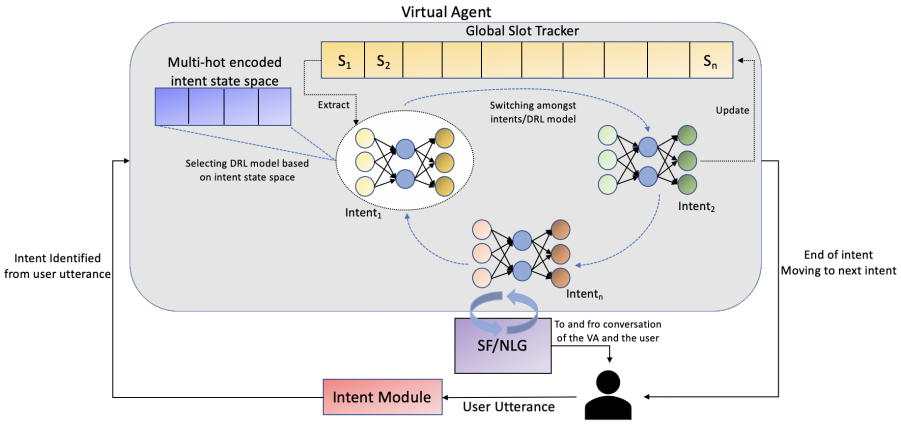


Fig. 1: Flow diagram for the proposed system : S_1,....,S_n refer to different slots present in the system

instead of training a single DRL network, we train a set of networks, where each network represents servicing a particular intent to complete a particular sub-dialogue. In the context of this work, a sub-dialogue refers to a conversation serving a single intent. Each DRL network exhibits unique behavior while interacting with the user to complete its task optimally. The flow diagram of the proposed system is shown in Figure 1.

Therefore, in a network of Deep Reinforcement Agents, an action selection is performed based on an optimal policy which is :

$$\pi^*_{\theta^{(i)}}(s) = argmax_{a \in A^{(i)}} Q^{*(i)}(s, a; \theta^{(i)}), \tag{5}$$

where intent $i \in I$ ($I \in \{\text{set of intents}\}$), is determined based on :

$$i = argmax_{i' \in I} G(i'|f), \tag{6}$$

where feature f comprises of all the features specifically the word vectors that are used to represent the user utterance. Equation 6 is used for high-level hierarchy, i.e., intent transition and Equation 5 is used for low-level hierarchy

i.e., within a DRL model transition. The inputs to the DRL model is obtained from

$$\overrightarrow{y} = H(f) \tag{7}$$

where $\overrightarrow{y}$ gives the slot tags for each of the words of the user utterance as seen in Table 1. Therefore, function G and H are pre-requisites for Network of Deep Reinforcement Learners where G and H represents the IC and SF module respectively. Due to lack of good quality conversational data for multiple intents of a particular domain and the effort involved in training a RL agent, we have developed an environment that is based on a pseudo-random generator to mimic the confidence values and output from the SF and IC module respectively. This environment and training procedure is curated to represent a real SF and IC as closely as possible and expedite the process of training to be much faster and robust to random noises that might exist in a NLU module.

Global Slot Tracker One of the elementary challenges in developing a multi-intent dialogue system is to ensure certain inter-subtask constraints which we refer to as slot constraints. This is rather intuitive as various intents of a domain share overlapping information amongst each other. Thus, eliciting these information from the user multiple times in a conversation individually for each intent makes the dialogue redundant and increases user dissatisfaction. To counter this issue, we maintain a global slot tracker for all the slots of a domain. Thus, we now have a global slot tracker which manages information of all the slots across all the intents of a domain. These information are updated in input state S_2 at every time step.

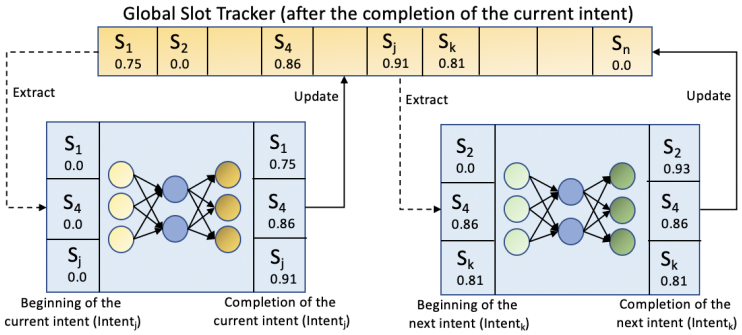


Fig. 2: State Space Update using Global Slot Tracker

State Space Update During training of the networks, the following procedure is adopted for the elicited information to be shared across multiple intents :

- State S_1 is initialized randomly invoking a DRL model corresponding to that particular intent.
- Based on the model invoked, the state S_2 of the intent is initialized randomly mimicking the confidence values of the SF module. The Q-learning updates are then applied to that individual DRL model only.
- After the current DRL model terminates, the state S_1 changes to invoke another DRL model corresponding to the intent represented by S_1.
- During intent transition, the overlapping slots of the previous intent are transferred to the current intent.
- For slot transfer, the Dialogue Manager keeps track of all the slots filled that are present in the system using the Global Slot Tracker. This allows the current DRL model to fetch the common values of the slot already filled by another DRL model.
- Therefore, for the current intent, its S_2 value is initialized by the information obtained from the previous DRL model, if any. Otherwise it is initialized randomly (as described earlier) so the VA can pick up actions given any dialogue state. This reduces unnecessary number of turns to serve the user and to avoid eliciting redundant information if any. State space update using Global slot tracker is illustrated in Figure 2.

Global Slot Tracker One of the elementary challenges in developing a multi-intent dialogue system is to ensure certain inter-subtask constraints which we refer to as slot constraints. This is rather intuitive as various intents of a domain share overlapping information amongst each other. Thus, eliciting these information from the user multiple times in a conversation individually for each intent makes the dialogue redundant and increases user dissatisfaction. To counter this issue, we maintain a global slot tracker for all the slots of a domain. Thus, we now have a global slot tracker which manages information of all the slots across all the intents of a domain. These information are updated in input state S_2 at every time step.

Training and Testing The system is trained on a pseudo-environment mimicking the confidence values of the SF module as an input to the state S_2 of different intents. This makes the trained DM module robust towards the performance of the NLU module and gives it the flexibility to generalize to any other state space since it has not been trained on a particular corpus or conversational data for a task, thereby prohibiting it to learn features and policies specific to a corpus. The procedure to train the proposed system is presented in Algorithm 1. Later, the learned policy which is trained on the pseudo-environment is tested against real IC and SF modules. Thus, real IC and SF modules are integrated with the system replacing the randomness from states S_1 and S_2, thereby incorporating natural language to test the robustness of the policy learnt. The rest of the system functions exactly in the same manner as described during training exhibiting state transfer information, optimally completing the sub-dialogues for a successful dialogue conversation.

Algorithm 1 Network of Deep-Q Learners

Initialize : Set of Deep-Q-Networks with replay memories $M^{(i)}$, action-value functions $Q^{(i)}$ with random weights $\theta^{(i)}$ and target action value functions $\widehat{Q}^{(i)}$ with weights $\widehat{\theta}^{(i)} = \theta^{(i)}$, Global Slot Tracker GST
repeat
  $S_1 \leftarrow i \leftarrow$ initial intent, predefined or determined by $argmax_{i \in I} G_0(I)$
  initial environment state in $S_2^{(i)}$
  repeat
    repeat
      Choose action $a \in A^{(i)}$ in S_2 derived from $Q^{(i)}$ ▷ e.g. ϵ greedy
      Execute action a and observe reward r and next state S_2'
      Append transition (S_2, a, r, S_2') to $M^{(i)}$ with maximal priority $P_t = max_{j<t} P_j$
      $E^{(i)} \leftarrow$ sample random mini-batch of experiments from $M^{(i)}$ based on maximum priority

$$y_k = \begin{cases} r_k, & \text{if final step of episode} \\ r_k + \gamma max_{a \in A^{(i)}} \widehat{Q}^{(i)}(S_2', a'; \theta^{(i)}), & \text{otherwise} \end{cases}$$

      Update transition priority $P_j = |y_k|$
      Gradient descent step on $\left(y_j - Q^{(i)}(S_2', a'; \theta^{(i)})\right)^2$ using $E^{(i)}$
      Reset $\widehat{Q}^{(i)} = Q^{(i)}$ every C steps
      $S_2 \leftarrow S_2'$
    until a is the terminating action, update GST
    $i' \leftarrow$ select next intent according to $argmax_{i' \in I} G(i'|f)$ ▷ f represents the features of the user utterance
    $S_1 \leftarrow i \leftarrow i'$
    $S_2 \leftarrow$ transfer slots from GST with intent i.
  until no intent change, initialize GST
until convergence ▷ Given number of Episodes completed

4.3 Implementation

This section describes the implementation details of the system including the *NLU* that comprises of Intent Classification and Slot-Filling Module, a template based Natural Language Generation (NLG) framework etc. for the development of End-to-end Task-oriented Multi-intent System.

Intent Classification Module The task of this module is to identify or predict one or more of the intents from the user's utterance. Thus, its objective is to maximize the conditional probability of intent(s) i given x.

$$P(i|x) = \prod_{q=1 \ldots n} P(i_q|x) \tag{8}$$

where n represents the number of intents in a domain. For this, a two layer CNN based deep learning model has been trained on the ATIS dataset. The input to the network is the word embeddings of the corresponding words in the utterance. GloVe word embedding [23] of dimension 300 has been used to represent words. Softmax activation has been used at the final layer for classification. We obtain a classification accuracy of 89% based on this model.

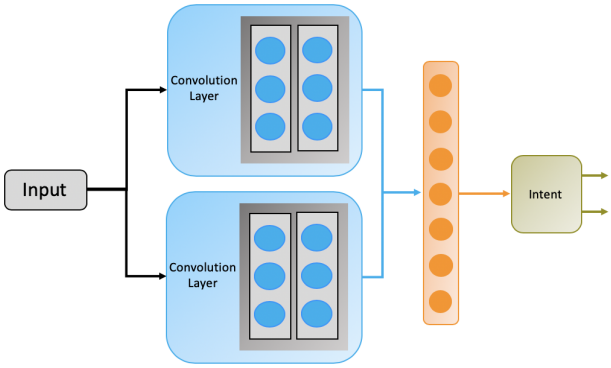


Fig. 3: An Architectural Diagram of Intent Classification Module

Thus, this module identifies one of the five intents which is the input to state S_1 of the state space.

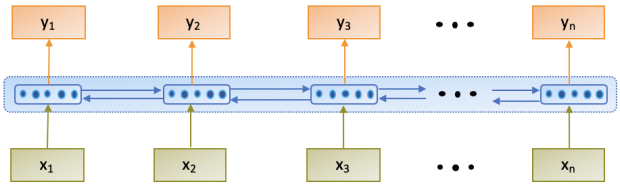


Fig. 4: An Architectural Diagram of Slot-Filling Module

Slot-Filling Module To extract relevant information from the user's utterance, an SF module has been trained. It is a deep learning model which uses a single Bi-directional Long Short Term Memory (LSTM) Network [11] at its core with the softmax activation at the final layer.

$$\mathbf{y} = Bi - LSTM(\mathbf{x}) \tag{9}$$

where $\mathbf{x}$ is the input word sequence and $\mathbf{y}$ contains its corresponding slot labels. This module is trained on the same ATIS dataset, wherein only the five intents mentioned above as per the dataset have been used with a fixed number of slots. The developed model achieved an accuracy of 86%. The necessary slots identified, along with the probability scores of the predicted labels are used by state S_2 for further processing. The slot labels utilized from the ATIS dataset for different intents for this particular work are listed in Table 6.

Table 6: List of ATIS slot labels that have been used to demonstrate the Air Travel domain for different intents

(a)

Air Travel Slots	deptCity	arrCity	deptTime	depDay
ATIS Slots Used	B-fromloc.city_name, B-fromloc.state_code, B-fromloc.state_name, I-fromloc.city_name, I-fromloc.state_name	B-toloc.city_name, B-toloc.country_name, B-toloc.state_code, B-toloc.state_name, I-toloc.city_name, I-toloc.state_name	B-depart_time.end_time, B-depart_time.period_mod, B-depart_time.period_of_day, B-depart_time.start_time, B-depart_time.time, I-depart_time.end_time, I-depart_time.period_of_day, I-depart_time.start_time, I-depart_time.time, I-depart_time.time_relative	B-depart_date.day_name, B-depart_date.day_number, B-depart_date.month_name, B-depart_date.year, I-depart_date.day_number, I-depart_date.today_relative
Intent	flight, airfare, airline	flight, airfare, airline	flight	flight, airfare

(b)

Air Travel Slots	class	RTrip	Gcity	TranType
ATIS Slots Used	B-booking_class, B-class_type, B-economy, I-class_type, I-economy	B-round_trip, I-round_trip	B-city_name, I-city_name	B-transport_type, I-transport_type
Intent	flight, airfare, airline	airfare	ground service, ground fare	ground service, ground fare

Natural Language Generation A retrieval based NLG framework has been used that maps the action picked up by the VA to its corresponding natural language to present to the user. Similarly, predefined sentence templates with slot placeholders which are replaced by the user goal for a dialogue have been defined for the user responses to present to the VA [8]. For e.g., an action picked up by the VA *askDate*, is mapped to multiple sentences pertaining to that action, out of which one is picked up in random. The randomly picked up sentence pertaining to an action is replaced by slot placeholders (if any) and presented to the user. Similarly, for the user, a pre-defined user goal is initialized at the start of the dialogue which remains intact throughout the entire conversation. Based on system's or VA's action *askDate* (let's say), an action is mapped for the user which is *replyDate* (let's say). This in turn is mapped to multiple sentences pertaining to the response of the user that is picked up in random, substituted with necessary slot placeholders (if any such as user goal) and presented to the VA for further processing.

Model Architecture The architecture of the neural network is as follows: Number of nodes in the input layer is equivalent to the size of the state vector S_2 for each intent, followed by one hidden layer with 75 nodes. Number of nodes in the output layer is equivalent to the action set of individual intent being served. The activation function of the hidden layer is Rectified Linear Units to normalize their weights. The DDQN-PER algorithm is used as the learning algorithm. Several variants of DRL based Q-learning algorithm were also employed. such as Deep Q-Network (DQN) [21], DDQN [30], DQN-PER and

Table 7: The final set of chosen hyper-parameter values

Hyper-parameter	Value
discount factor (γ)	0.7
minimum epsilon	0.15
experience replay size	100000
activation function	Rectified Linear Unit
batch size	32
weights (action value functions)	random

DDQN-PER. It was observed that DDQN-PER performed the best amongst all other learning algorithms in terms of convergence [25]. The other parameters of the model are : discount factor (γ) = 0.7 [8], [5] minimum epsilon = 0.15, experience replay size = 100000 [8], batch size = 32 [8]. The training is done for 50000 dialogues. Also, the hyper-parameters of the model were assigned through thorough literature survey and careful analysis as follows : Setting higher value of discount factor was not resulting in the proper convergence of the algorithm, also lower epsilon values were not sufficiently exploring the state spaces, causing the agent to get stuck in local optima. Please note that we use all standard hyper-parameter values from the literature for the convergence of the policy. We do not make any explicit attempt to tune the algorithm with other values. The list of hyper-parameter values used for training is listed in Table 7.

5 Results and Discussion

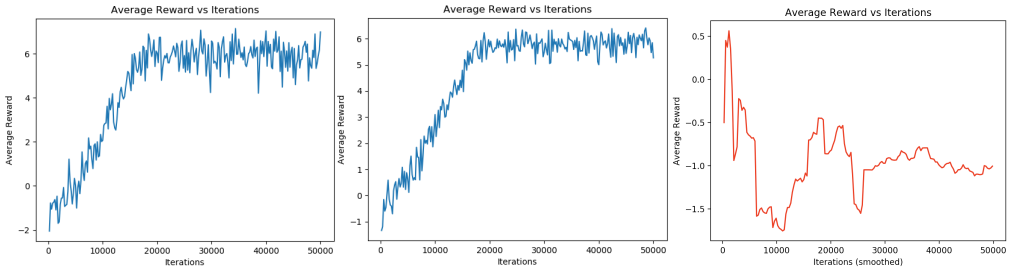
The following metrics were used to analyze the performance of various baselines and the proposed framework :

1. *Average episodic reward* : The cumulative reward through all the time-steps at the end of a dialogue. Higher the cumulative reward, better is the chances of task-completion.
2. *Average dialogue length* : It is basically the average system actions per dialogue. The VA should be able to complete its task in less number of time-steps.
3. *Training time*[1] : It gives an estimate of the computational requirement of different VAs.
4. *Human evaluation* : It gives an estimate of the qualitative analysis of the conversations.

For a given dialogue conversation, the task of the VA is to get the maximum cumulative reward at the end of the episode and it should be able to do it in lesser number of turns. These values are computed by taking the average of 100 such executions of the policy learnt with the intents picked up in random. Here, we present a comparison of the proposed multi-intent dialogue system

[1] Ran on Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz, 251GB RAM

with two baselines. While the two baselines use a single policy training for learning a strategy, the proposed system uses multi-policies.



(a) (b) (c)

Fig. 5: Learning Curves of the Agent for the individual DRL model and the flat DRL : (a) with five slots namely "flight" and "airfare", (b) with two slots namely "ground service" and "ground fare", (c) for the flat DRL agent (first baseline)

5.1 Comparison with the Baselines

To evaluate the performance of the proposed framework, we compare our model with the following baselines :

- **First baseline** : Learns a single policy for multiple intents where the state space comprises of all the slots of a domain without any distinction of slot types amongst different intents i.e., a flat DRL agent.
- **Second baseline** : Learns distinct policy for individual intent, but the overlapping information or the state space is not shared amongst various models, thus making the VA redundant in filling the same slots which are common amongst various intents, thereby increasing the number of turns to serve the user. The values reported for the second baseline are obtained by considering the mean of the values obtained by executing different intents sequentially (in the same order as the proposed system).

Table 8: Quantitative Analysis of the proposed hierarchical multi-intent system. † signifies that all the obtained results are statistically significant with respect to baselines

Number of intents served	**1**	**2**	**3**	**4**	**5**
Average Episodic Reward	41.21 ± 16.79†	81.26 ± 18.13†	111.49 ± 16.74†	140.03 ± 14.44†	166.72 ± 9.55†
Average Dialogue Length	5.45 ± 1.99†	10.16 ± 2.23†	13.22 ± 1.76†	15.43 ± 1.33†	17.21 ± 1.15†
Training Time (in hrs)	5.50				

Table 9: Quantitative Analysis of the baseline systems

	First baseline	Second baseline			
	Single Intent	No. of intents served			
		2	3	4	5
Average Episodic Reward	-171.28 ± 30.05	88.89 ± 23.35	132.91 ± 21.45	173.45 ± 16.83	214.84 ± 10.20
Average Dialogue Length	59.84 ± 3.7	12.44 ± 2.99	17.83 ± 2.60	23.62 ± 2.20	29.13 ± 1.59
Training Time (in hrs)	10.55	−−			

Figures 5a and 5b show the learning curves for the individual models of the proposed multi-intent dialogue system. It is evident from the overall nature of the graphs that the learning curve is linearly increasing with the number of iterations and then saturates after some time when it learns an efficient policy with little fluctuations. Figure 5c shows the learning curve for the first baseline i.e., the flat DRL system. It is observed from the curve that the mono-policy baseline did not improve with time. This is supposedly because of the abstraction exhibited in the hierarchical approach where dedicated system actions for specific tasks are available in different DRL models rather than knitting them across multiple intents. Thus, the increased complexity in the flat DRL agent owing to flat state space and the lack of focused system actions to handle multiple intents of the user prevents it from learning an effective dialogue policy.

From Table 8 and Table 9, it is observed that the proposed system performs better than the first baseline for servicing single intent by significant margin in terms of average dialogue length. Since the first baseline is agnostic of intents, the VA tries to acquire information for all the slots often tending to ask irrelevant questions to the user that might not be in context with the users' current need whereas the proposed system benefits from the abstraction exhibited in the multi-policy approach. It is also evident that training the first baseline was two-fold harder than training the proposed system.

The second baseline on average takes more number of dialogue turns to attain the same amount of information as compared to the proposed model. Not having the capability to transfer information about slots from one intent to the other, the systems ask for overlapping slots tending to make the conversation trite and redundant. The proposed system attains an improvement of **41%** in terms of dialogue length for servicing 5 intents compared to second baseline. Figure 6 shows the verbalization of the policy learnt by the proposed dialogue system. Verbalization of the policy for the second baseline system which serves the same intents that are being served by the proposed system are shown in Figure 7a and 7b. It is indeed evident from the chat transcripts that the proposed system services the multiple intents of the system in a more efficient and practical manner rather than the second baseline system that takes unnecessary turns (4 more) for eliciting redundant information for servicing each intents separately.

System Log : No intent
S_1 : [0 0 0 0 0]
Agent > *"Hello How can I help you ?"*
User > *"I would like to travel from pittsburgh to denver"*
System Log : Intent Changed to 'flight'
S_1 : [1 0 0 0 0]
S_2 : [0.98 0.97 0. 0. 0.]
Agent [askClass] > *"Please specify the class of flight"*
User > *"I want to travel in economy class"*
S_1 : [1 0 0 0 0]
S_2 : [0.98 0.97 0. 0. 0.94]
Agent [askDateTime] > *"Specify the time and date"*
User > *"On fourth january at morning"*
S_1 : [1 0 0 0 0]
S_2 : [0.98 0.97 0.87 0.99 0.94]
Agent [closing_conversation] > *"The flights from pittsburgh to denver on fourth january at morning via economy class are JW456, GF678.*
Do you need anything else ?"
User [Yes/No] > *"Show me the cheapest fares"*
System Log : Intent Changed to 'airfare'
S_1 : [0 1 0 0 0]
S_2 : [0.98 0.97 0. 0.87 0.94]
Agent [askRtrip] > *"Round trip or one way fare ?"*
User > *"round trip"*
S_1 : [0 1 0 0 0]
S_2 : [0.98 0.97 0. 99 0.87 0.94]
Agent [closing_conversation] > *"The cheapest fare of the flight from pittsburgh to denver on fourth january and round trip via economy class is $1000 for JW456.*
Do you need anything else ?"
User [Yes/No] > *"No Thanks"*
System Log : No Intent
Exit

Fig. 6: Verbalization of the Policy learnt for the proposed multi-intent system

System Log : Flight intent
Agent > *"Welcome to flight enquiry system"*
User > *"Show me flights to orlando"*
S_2 : [0. 0.98 0. 0. 0.]
Agent [askdeptCity] > *"Specify the departure city"*
User > *"I want to travel from washington"*
S_2 : [0.97 0.98 0. 0. 0.]
Agent [askClass] > *"Please specify the class of flight"*
User > *"business"*
S_2 : [0.97 0.98 0. 0. 0.99]
Agent [askDateTime] > *"Specify the time and date"*
User > *"Need to travel on twenty eight december at morning"*
S_2 : [0.97 0.98 0.94 0.98 0.99]
Agent [closing_conversation] > *"The flights from washington to orlando on twenty eight december at morning via business class are TN456, AU678.*
Thanks for using the flight attendant."
System Log : Exit

(a) for *flight* intent

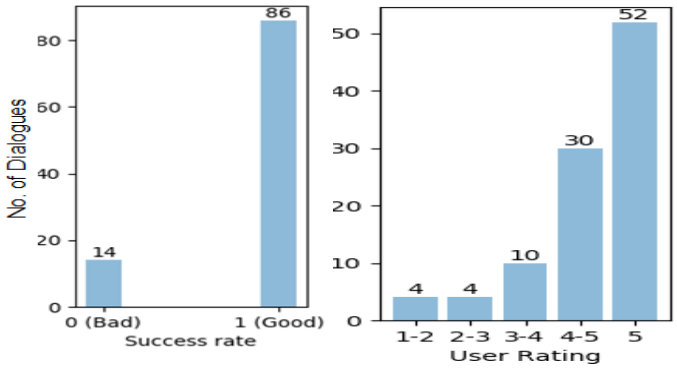
System Log : Airfare intent
Agent > *"Hello How can I help you ?"*
User > *"Show me the cheapest fares"*
S_2 : [0. 0. 0. 0. 0.]
Agent [askDeptArr] > *"From where to where ?"*
User > *"I want to travel from washington to orlando"*
S_2 : [0.97 0.98 0. 0. 0.]
Agent [askClass] > *"Please specify the class of flight"*
User > *"I want to travel in business class"*
S_2 : [0.97 0.98 0. 0. 0.99]
Agent [askDateRtrip] > *"Specify the date and whether round trip or one way fare ?"*
User > *"Need to travel on twenty eight december and one way trip fare"*
S_2 : [0.97 0.98 0.99 0.42 0.99]
Agent [reaskdepDay] > *"Are you travelling on twenty eight december ?"*
User [yes/no] > *"yes"*
S_2 : [0.97 0.98 0.99 1 0.99]
Agent [closing_conversation] > *"The cheapest fare of the flight from washington to orlando on twenty eight december and one way trip via business class is $1100 for TN456.*
Thanks for using the flight attendant."
System Log : Exit

(b) for *airfare* intent

Fig. 7: Verbalization of the Policy learnt for second baseline system

5.2 Human Evaluation

Three human users from the authors' affiliation were presented with 100 dialogues to rate the general quality on two marking schema: **(i)** Rating the



(a) (b)

Fig. 8: Performance of the VA tested with human evaluators : (a) success rate based on binary marking schema, (b) Distribution of user-ratings based on variable marking schema

dialogue on a scale of 1 (worst) to 5 (best) to get a detailed marking score based on *coherence*, *ease of answering* and *naturalness*. By *coherence*, we mean that the VA should ask questions or provide information based on the query of the user. The users' need and the VAs' actions should be coherent. *Ease of answering* refers to how effortless it was for the user to communicate the information and its need to the VA at every dialogue turn collectively. *Naturalness* refers to the users' view on how suitable or successful the VA can be in its endeavor without much difficulty in terms of achieving the user goal. **(ii)** Binary scoring of 1 (good) or 0 (bad) to evaluate whether the conversation was successful or not. Figure 8b shows the subjective evaluation in terms of user rating based on the first marking schema. For the binary marking schema, a particular conversation was deemed as successful based on the voting schema, if two or more judges agreed on the same. Figure 8a presents the performance of the VA against human evaluators in terms of the success rate.

5.3 Error Analysis

Detailed observation and analysis of the proposed system revealed various scenarios where the system falters which are discussed as follows :

- **Intent Identification Error :** S_1 is managed by the IC module which in turn provides input to the DRL model. A misclassification of the intended intent due to the limitation of the IC module leads to the Dialogue Manager serving a wrong intent. For eg. for an user utterance "*how much does the limousine service cost?*", the intent was incorrectly identified as "ground service" instead of "ground fare", that leads to the Dialogue Manager executing a wrong DRL model based on the input from the S_1 state space; thus, making the user dissatisfied by the information provided by the VA.

– **Slot-filling Error :** Similarly, a mis-identification of the relevant user information in the form of slots leads to the VA taking extra turns to retrieve the correct information thereby increasing dialogue length. For eg., for an user utterance "*I want to travel on fifteenth january*", the **deptDay** slot was wrongly identified as "january "with a very low confidence. This prompted the VA to confirm the acquired slot from the user as per its policy, to which the user denied, thereby taking extra turns to elicit correct information from the user.

5.4 Comparison with State of the Art

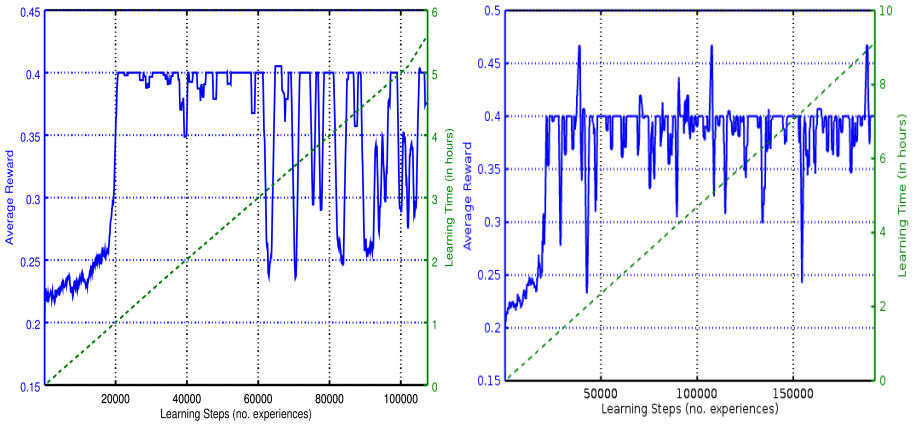
Here, we present a comparison with the state of the art relevant to the context of our proposed setting. In this particular work [5], the authors present a VA in the domain of restaurant. Thus, we apply the same model here for the "Air-Travel" domain and the results are reported in Table 10. Figure 9a and 9b shows the learning curve of the state of the art [5] system for the "flight" and "airfare" intent respectively. It is a rather unstable curve with numerous rise and fall in the learning pattern. Close observation of the system revealed that the agent tends to get stuck in a loop thus, executing an action for a number of times in a sequence. The average dialogue length to serve the user for the "flight" intent was **17.75**. Whereas the same intent was served by the proposed system in **6.44** turns as seen in table 10. Figure 10 shows the verbalization of the policy learnt by the state of the art system for the "flight" intent. It is indeed evident from the chat transcripts and Table 10 that the proposed system services the intents of the system in a lesser no.of turns rather than the state of the art system that takes numerous turns for eliciting same information for servicing each intents.

However, the current work also has certain limitations which are as follows :

– Because of absence of high quality dialogue data, the training of the VA has been done using a pseudo-environment, i.e., simulator. However, training the VA with real data has the capability to make it much more diverse and pertinent.
– Also, currently the VA is incapable of handling certain scenarios such as unknown or a dynamic slot entity communicated by the user through the course of the dialogue. This can happen in situations where a rare slot not known to the VA has been informed by the user. So, the VA manages such a scenario in a restricted way (say lesser user contentment), as ideally it is unequipped with an efficient error-handling strategy in that aspect.
– Concepts such as Transfer Learning has been unexplored in the current work. These directions need to be addressed to manage situations such as less availability of data for a domain compared to the other. The simulator based training assumes that there are equal amount of data for training different domains in a balanced manner. However, in much practical scenario, this argument needs to be addressed.

Table 10: Comparison with the State of the Art for different intents

	Proposed Model	State of the Art
	Average Dialogue Length	
Intent with 5 slots (namely flight)	6.44 ± 2.09	17.75
Intent with 3 slots (namely airline)	4.55 ± 1.43	10.83
Intent with 2 slots (namely ground service)	3.1 ± 0.9	7.56



(a) for *flight* intent (b) for *airfare* intent

Fig. 9: Learning Curves of the State of Art Systems

6 Conclusions and Future Work

This paper proposes a hierarchical method for learning an efficient DM strategy for a multi-intent dialogue system. It uses a network of Deep Reinforcement Agents (DDQN-PER here), to train agents that specialize in serving specific intents and share overlapping information within the network as slots of different intents in a domain are not discrete and are dependent on each other. Results show that the developed system is indeed scalable and is a practical solution towards serving multiple intents of the user with consistency, thereby, restricting the VA to extract redundant information.

In future, efficient optimization of the learning algorithm can be employed to improve the convergence of the policy. Other future works include administering and improving the proposed system to cover larger sets of intents across varied domains with minimal changes and effort. Also, there exists a myriad of works that are focused on creating VAs in prevalent languages for which conversational data is available in abundance such as *English*. However, there is an explicit dearth of works that propose VAs for low-resource languages even though it is used extensively by the masses primarily because of unavailability of conversational data, various NLP resources and tools in the required language. Thus, creating VAs in low-resource languages is a direction that

Fig. 10: Verbalization of the policy learnt by the state of the art system for intent "flight"

needs to be dealt upon seriously. Other behavioral information apart from user semantics such as sentiment and multi-modal aspects should be taken into account for policy learning in order to attain maximum user satisfaction.

References

1. Arulkumaran, K., Deisenroth, M.P., Brundage, M., Bharath, A.A.: A brief survey of deep reinforcement learning. arXiv preprint arXiv:1708.05866 (2017)
2. Bordes, A., Boureau, Y.L., Weston, J.: Learning end-to-end goal-oriented dialog. arXiv preprint arXiv:1605.07683 (2016)
3. Budzianowski, P., Ultes, S., Su, P., Mrksic, N., Wen, T., Casanueva, I., Rojas-Barahona, L.M., Gasic, M.: Sub-domain modelling for dialogue management with hierarchical reinforcement learning. In: Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, Saarbrücken, Germany, August 15-17, 2017, pp. 86–92 (2017). DOI 10.18653/v1/w17-5512. URL https://doi.org/10.18653/v1/w17-5512
4. Casanueva, I., Budzianowski, P., Su, P., Ultes, S., Rojas-Barahona, L.M., Tseng, B., Gasic, M.: Feudal reinforcement learning for dialogue management in large domains. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers), pp. 714–719 (2018). URL https://aclanthology.info/papers/N18-2112/n18-2112
5. Cuayáhuitl, H.: Simpleds: A simple deep reinforcement learning dialogue system. In: Dialogues with Social Robots, pp. 109–118. Springer (2017)
6. Cuayáhuitl, H., Keizer, S., Lemon, O.: Strategic dialogue management via deep reinforcement learning. arXiv preprint arXiv:1511.08099 (2015)
7. Cuayáhuitl, H., Yu, S., Williamson, A., Carse, J.: Deep reinforcement learning for multi-domain dialogue systems. arXiv preprint arXiv:1611.08675 (2016)

8. Cuayáhuitl, H., Yu, S., et al.: Deep reinforcement learning of dialogue policies with less weight updates (2017)
9. Fazel-Zarandi, M., Li, S.W., Cao, J., Casale, J., Henderson, P., Whitney, D., Geramifard, A.: Learning robust dialog policies in noisy environments. arXiv preprint arXiv:1712.04034 (2017)
10. Fraser, N.: Assessment of interactive systems. In: Handbook of standards and resources for spoken language systems, pp. 564–615. Mouton de Gruyter (1998)
11. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
12. Ilievski, V., Musat, C., Hossmann, A., Baeriswyl, M.: 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., pp. 4115–4121 (2018). URL https://doi.org/10.24963/ijcai.2018/572
13. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. Journal of artificial intelligence research **4**, 237–285 (1996)
14. Keizer, S., Guhe, M., Cuayáhuitl, H., Efstathiou, I., Engelbrecht, K.P., Dobre, M., Lascarides, A., Lemon, O.: Evaluating persuasion strategies and deep reinforcement learning methods for negotiation dialogue agents. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, vol. 2, pp. 480–484 (2017)
15. Levin, E., Pieraccini, R., Eckert, W.: Using markov decision process for learning dialogue strategies. In: Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on, vol. 1, pp. 201–204. IEEE (1998)
16. Li, X., Chen, Y.N., Li, L., Gao, J., Celikyilmaz, A.: End-to-end task-completion neural dialogue systems. arXiv preprint arXiv:1703.01008 (2017)
17. Lipton, Z., Li, X., Gao, J., Li, L., Ahmed, F., Deng, L.: Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
18. McTear, M.F.: Modelling spoken dialogues with state transition diagrams: experiences with the cslu toolkit. In: Fifth International Conference on Spoken Language Processing (1998)
19. McTear, M.F.: Spoken dialogue technology: enabling the conversational user interface. ACM Computing Surveys (CSUR) **34**(1), 90–169 (2002)
20. Meng, T.L., Khushi, M.: Reinforcement learning in financial markets. Data **4**(3), 110 (2019). DOI 10.3390/data4030110. URL https://doi.org/10.3390/data4030110
21. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602 (2013)
22. Peng, B., Li, X., Li, L., Gao, J., Çelikyilmaz, A., Lee, S., Wong, K.: 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, pp. 2231–2240 (2017). URL https://aclanthology.info/papers/D17-1237/d17-1237
23. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532–1543 (2014)
24. Price, P.J.: Evaluation of spoken language systems: The atis domain. In: Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990 (1990)
25. Saha, T., Gupta, D., Saha, S., Bhattacharyya, P.: Reinforcement learning based dialogue management strategy. In: L. Cheng, A.C.S. Leung, S. Ozawa (eds.) Neural Information Processing, pp. 359–372. Springer International Publishing (2018)
26. Schaul, T., Quan, J., Antonoglou, I., Silver, D.: Prioritized experience replay. arXiv preprint arXiv:1511.05952 (2015)
27. Serban, I.V., Sordoni, A., Bengio, Y., Courville, A.C., Pineau, J.: Building end-to-end dialogue systems using generative hierarchical neural network models. In: AAAI, vol. 16, pp. 3776–3784 (2016)

28. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction, vol. 1. MIT press Cambridge (1998)
29. Tang, D., Li, X., Gao, J., Wang, C., Li, L., Jebara, T.: Subgoal discovery for hierarchical dialogue policy learning. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018, pp. 2298–2309 (2018). URL https://aclanthology.info/papers/D18-1253/d18-1253
30. Van Hasselt, H., Guez, A., Silver, D.: Deep reinforcement learning with double q-learning. In: AAAI, vol. 16, pp. 2094–2100 (2016)
31. Wen, T.H., Vandyke, D., Mrksic, N., Gasic, M., Rojas-Barahona, L.M., Su, P.H., Ultes, S., Young, S.: A network-based end-to-end trainable task-oriented dialogue system. arXiv preprint arXiv:1604.04562 (2016)
32. Xu, P., Sarikaya, R.: Exploiting shared information for multi-intent natural language sentence classification. In: INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association, Lyon, France, August 25-29, 2013, pp. 3785–3789 (2013). URL http://www.isca-speech.org/archive/interspeech_2013/i13_3785.html
33. Zhao, T., Eskenazi, M.: Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. arXiv preprint arXiv:1606.02560 (2016)