SimpleDS: A Simple Deep Reinforcement Learning Dialogue System

Heriberto Cuayáhuitl[1]

Abstract This paper presents *SimpleDS*, a simple and publicly available dialogue system trained with deep reinforcement learning. In contrast to previous reinforcement learning dialogue systems, this system avoids manual feature engineering by performing action selection directly from raw text of the last system and (noisy) user responses. Our initial results, in the restaurant domain, report that it is indeed possible to induce reasonable behaviours with such an approach that aims for higher levels of automation in dialogue control for intelligent interactive agents.

1 Introduction

Almost two decades ago, the (spoken) dialogue systems community adopted the Reinforcement Learning (RL) paradigm since it offered the possibility to treat dialogue design as an optimisation problem, and because RL-based systems can improve their performance over time with experience. Although a large number of methods have been proposed for training (spoken) dialogue systems using RL, the question of "How to train dialogue policies in an efficient, scalable and effective way across domains?" still remains as an open problem. One limitation of current approaches is the fact that RL-based dialogue systems still require high-levels of human intervention (from system developers), as opposed to automating the dialogue design. Training a system of this kind requires a system developer to provide a set of features to describe the dialogue state, a set of actions to control the interaction, and a performance function to reward or penalise the action-selection process. All of these elements have to be carefully engineered in order to learn a good dialogue policy (or policies). This suggests that one way of advancing the state-of-the-art in this field is by reducing the amount of human intervention in the dialogue design process through higher degrees of automation, i.e. by moving towards truly autonomous learning.

[1]Computer Science at Heriot-Watt University, Edinburgh, Scotland, e-mail: hc213@hw.ac.uk

Recent advances in machine learning have proposed machine learning methods as a way to reduce human intervention in the creation of intelligent agents. In particular, the field of Deep Reinforcement Learning (DRL) targets feature learning and policy learning simultaneously—which reduces the effort in feature engineering [1]. This is relevant because the vast majority of previous RL-based dialogue systems make use of carefully engineered features to represent the dialogue state [2].

Motivated by the advantages of DRL methods over traditional RL methods, in this paper we present an extended dialogue system, recently applied to strategic dialogue management[3], that makes use of raw noisy text—without any engineered features to represent the dialogue state. By using this representation, the dialogue system does not require a Spoken Language Understanding (SLU) component. We bypass SLU by learning dialogue policies directly from (simulated) speech recognition outputs. The rest of the paper describes a proof of concept system in different languages (English, German and Spanish) which is trained based on this idea.

2 Deep Reinforcement Learning for Dialogue Control

A Reinforcement Learning (RL) agent learns its behaviour from interaction with an environment and the physical or virtual agents within it, where situations are mapped to actions by maximising a long-term reward signal [4]. An RL agent is typically characterised by: (i) a finite or infinite set of states $S = \{s_i\}$; (ii) a finite or infinite set of actions $A = \{a_j\}$; (iii) a state transition function $T(s,a,s')$ that specifies the next state s' given the current state s and action a; (iv) a reward function $R(s,a,s')$ that specifies the reward given to the agent for choosing action a in state s and transitioning to state s'; and (v) a policy $\pi : S \rightarrow A$ that defines a mapping from states to actions. The goal of an RL agent is to select actions by maximising its cumulative discounted reward defined as $Q^*(s,a) = \max_\pi \mathbb{E}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+1} + ...|s_t = s, a_t = a, \pi]$, where function Q^* represents the maximum sum of rewards r_t discounted by factor γ at each time step. While the RL agent takes actions with probability $Pr(a|s)$ during training, it takes the best actions $\max_a Pr(a|s)$ at test time.

To induce the Q function above we use Deep Reinforcement Learning as in [1], which approximates Q^* using a multilayer convolutional neural network. The Q function of a DRL agent is parameterised as $Q(s,a;\theta_i)$, where θ_i are the parameters (weights) of the neural net at iteration i. More specifically, training a DRL agent requires a dataset of experiences $D_t = \{e_1, ...e_t\}$ (also referred to as 'experience replay memory'), where every experience is described as a tuple $e_t = (s_t, a_t, r_t, s_{t+1})$. The Q function can be induced by applying Q-learning updates over minibatches of experience $MB = \{(s,a,r,s') \sim U(D)\}$ drawn uniformly at random from dataset D. A Q-learning update at iteration i is thus defined as the loss function $L_i(\theta_i) = \mathbb{E}_{MB}\left[(r + \gamma \max_{a'} Q(s',a';\overline{\theta}_i) - Q(s,a;\theta_i))^2\right]$, where θ_i are the parameters of the neural net at iteration i, and $\overline{\theta}_i$ are the target parameters of the neural net at iteration i. The latter are only updated every C steps. This process is implemented in the learning algorithm *Deep Q-Learning with Experience Replay* described in [1].

3 The *SimpleDS* Dialogue System

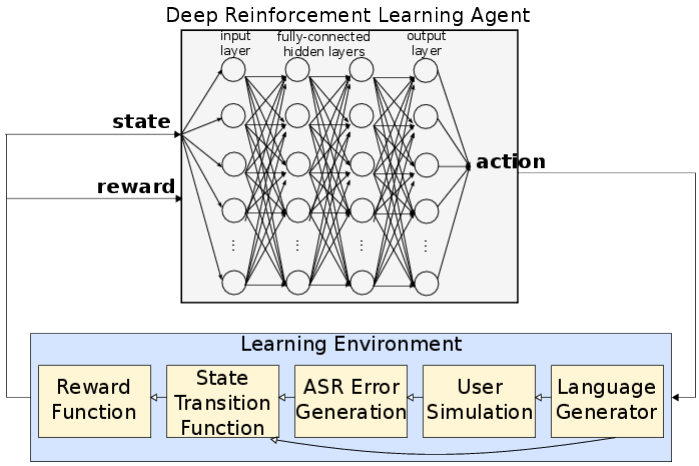

Fig. 1 High-level architecture of the *SimpleDS* dialogue system–see text for details.

Figure 1 shows a high-level diagram of the *SimpleDS* dialogue system. At the bottom, the learning environment receives an action (dialogue act) and outputs the next environment state and numerical reward. To do that, the environment first generates the word sequence of the last system action, the user simulator generates a word sequence as a response to that action, and the user response is distorted given some noise level and word-level confidence scores. Based on the system's verbalisation and noisy user response, the next dialogue state and reward are calculated and given as a result of having executed the given action. At the top of the diagram, a Deep Reinforcement Learning (DRL) agent receives the state and reward, updates its policy during learning, and outputs an action according to its learnt policy.

This system runs under a client-server architecture, where the environment acts as the *server* and the learning agent acts as the *client*. They communicate by exchanging messages, where the client tells the server the action to execute, and the server tells the client the dialogue state and reward observed. The *SimpleDS* learning agent is based on the ConvNetJS tool [5], which implements the algorithm 'Deep Q-Learning with experience replay' proposed by [1]. We extended this tool to support multi-threaded and client-server processing with constrained search spaces.[1]

The *state space* includes up to 100 word-based features depending on the vocabulary of the *SimpleDS* agent in the restaurant domain. The initial release of *SimpleDS* provides support for English, German and Spanish. While words derived from system responses are treated as binary variables (i.e. word present or absent), the words derived from noisy user responses can be seen as continuous variables by taking

[1] The code of *SimpleDS* is available at `https://github.com/cuayahuitl/SimpleDS`

confidence scores into account. Since we use a single variable per word, user features override system ones in case of overlaps.

The *action space* includes 35 dialogue acts in the Restaurant domain[2]. They include 2 salutations, 9 requests, 7 apologies, 7 explicit confirmations, 7 implicit confirmations, 1 retrieve information, and 2 provide information. Rather than learning with whole action sets, *SimpleDS* supports learning from constrained actions by applying Q-learning learning updates only on the set of valid actions. The constrained actions come from the most likely actions (e.g. $Pr(a|s) > 0.01$) with probabilities derived from a Naive Bayes classifier trained from example dialogues. The latter are motivated by the fact that a new system does not have training data apart from a small number of demonstration dialogues. In addition to the most probable data-like actions, the constrained action set is extended with the legitimate requests, apologies and confirmations in state s. The fact that constrained actions are data-driven and driven by application independent heuristics facilitates its usage across domains.

The *state transition function* is based on a numerical vector representing the last system and user responses. The former are straightforward, 0 if absent and 1 if present. The latter correspond to the confidence level [0..1] of noisy user responses. Given that *SimpleDS* targets a simple and extensible dialogue system, it uses templates for language generation, a rule-based user simulator, and confidence scores generated uniformly at random (words with scores under a threshold were distorted).

The *reward function* is motivated by the fact that human-machine dialogues should confirm the slots required (*CR*) and that they should be human-like (*DR*). It is defined as $R(s,a,s') = (CR \times w) + (DR \times (1-w)) - DL$, where *CR* is the number of positively confirmed slots divided by the slots to confirm; w is a weight over confirmations, we used w=0.5; *DR* is a probability of having observed action a in state s in the demonstrations, and *DL* is used to encourage efficient interactions, we used *DL*=0.1. The *DR* scores are derived from the same statistical classifier as above, which allows us to do statistical inference over actions given states ($Pr(a|s)$).

The *model architecture* consists of a fully-connected multilayer neural net with up to 100 nodes in the input layer (depending on the vocabulary), 40 nodes in the first hidden layer, 40 nodes in the second hidden layer, and 35 nodes (action set) in the output layer. The hidden layers use Rectified Linear Units to normalise their weights [6]. Finally, the learning parameters are as follows: experience replay size=10000, discount factor=0.7, minimum epsilon=0.01, batch size=32, learning steps=20000. A comprehensive analysis comparing multiple state representations, action sets, reward functions and learning parameters is left as future work.

Figure 2 shows the learning curve of a *SimpleDS* agent using 3000 simulated dialogues. This agent uses a smaller set of actions per state (between 4 and 5 actions)

[2] Actions: Salutation(greeting), Request(hmihy), Request(food), Request(price), Request(area), Request(food, price), Request(food, area), Request(price, area), Request(food, price, area), AskFor(more), Apology(food), Apology(price), Apology(area), Apology(food, price), Apology(food, area), Apology(price, area), Apology(food, price, area), ExpConfirm(food), ExpConfirm(price), ExpConfirm(area), ExpConfirm(food, price), ExpConfirm(food, area), ExpConfirm(price, area), ExpConfirm(food, price, area), ImpConfirm(food), ImpConfirm(price), ImpConfirm(area), ImpConfirm(food,price), ImpConfirm(food, area), ImpConfirm(price, area), ImpConfirm(food, price, area), Retrieve(info), Provide(unknown), Provide(known), Salutation(closing).

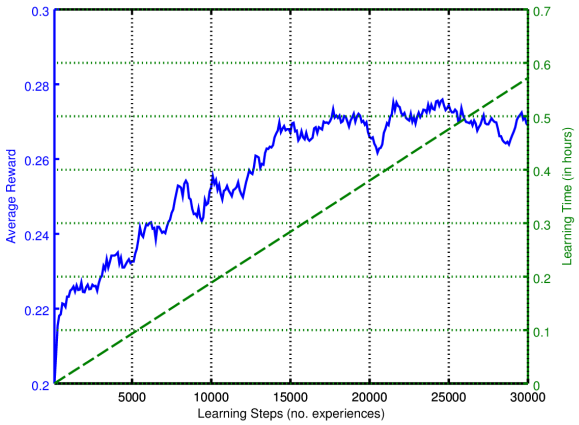

Fig. 2 Learning curve of the *SimpleDS* Deep Reinforcement Learning agent–see text for details.

rather than the whole action set per state—according to the application-independent heuristics mentioned in the previous Section. This reduction has the advantage that policies can be learnt quicker, that more sensible dialogues can potentially be learnt, and that it is inherent that some domains make use of legitimate actions during the interaction. In the case of more complex systems, with higher amounts of features and actions, learning with valid actions (rather than all actions) can make a huge difference in terms of computational resources and learning time. The quality of the learnt policies will depend on the learning environment and given constraints.

Table 1 shows an example dialogue of the learnt policy with user inputs derived from simulated speech recognition results. Our initial tests suggest that reasonable interactions can be generated using the proposed learning approach. *SimpleDS* has been demonstrated on a mobile App, and a human evaluation is left as future work.

4 Summary

We describe a publicly available dialogue system motivated by the idea that future dialogue systems should be trained with almost no intervention from system developers. In contrast to previous reinforcement learning dialogue systems, *SimpleDS* selects dialogue actions directly from raw (noisy) text of the last system and user responses. It remains to be demonstrated how far one can go with such an approach. Future work includes to (a) compare different model architectures, training parameters and reward functions; (b) extend or improve the abilities of the proposed dialogue system; (c) train deep learning agents in other (larger scale) domains [7, 8, 9]; (d) evaluate end-to-end systems with real users; (e) compare or combine different types of neural nets [10]; and (e) perform fast learning based on parallel computing.

Environment State	**Action**	**Verbalisation**
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	Salutation (greeting)	Hello!
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	Request (food,price,area)	What type of food, price range, and area are you looking for? [reasonably priced mexican ...food in the east of town]
0,0,0,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0.53,0,0,0,0,0.63,1,0,0,0,0,0,0,0.83,0,0,0,0,0,0,0,0,1,0,0, 0,0.78,0,0,0,0,0.31,0,0,0,1,0.36,1,0.25,0,0,0,0,0,0,0,0,0,0.57,0,0,0.82,1,0,0,0,0,1,0,0,0,0,0,1	ImpConfirm (food,price,area)	Okay, reasonably priced mexican food in the east.
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0, 0,0,0,1,0,0,0,0,0,1,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	Retrieve (info)	Let me see.
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0, 0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	Provide (known)	Restaurant X is an excellent choice. It is located in Y.
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0, 0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0,0	AskFor (more)	Anything else? [no]
0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0.72,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	Salutation (closing)	Okay, talk to you soon. Bye!
0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,1		

Table 1 Example dialogue using the policy from Fig.2, where states are numerical representations of the last system and noisy user inputs, actions are dialogue acts, and user resposes are in brackets

Acknowledgments

Funding from the European Research Council (ERC) project "STAC: Strategic Conversation" no. 269427 is gratefully acknowledged.

References

1. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning. In: NIPS Deep Learning Workshop. (2013)
2. Paek, T., Pieraccini, R.: Automating spoken dialogue management design using machine learning: An industry perspective. Speech Communication **50**(8-9) (2008)
3. Cuayáhuitl, H., Keizer, S., Lemon, O.: Strategic dialogue management via deep reinforcement learning. In: NIPS Deep Reinforcement Learning Workshop. (2015)
4. Szepesvári, C.: Algorithms for Reinforcement Learning. Morgan and Claypool Pub. (2010)
5. Karpathy, A.: ConvNetJS: A javascript library for training deep learning models. https://github.com/karpathy/convnetjs (2015)
6. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: ICML. (2010)
7. Cuayáhuitl, H., Renals, S., Lemon, O., Shimodaira, H.: Evaluation of a hierarchical reinforcement learning spoken dialogue system. Computer Speech & Language **24**(2) (2010)
8. Cuayáhuitl, H., Dethlefs, N.: Spatially-aware dialogue control using hierarchical reinforcement learning. TSLP **7**(3) (2011)
9. Cuayáhuitl, H., Kruijff-Korbayová, I., Dethlefs, N.: Nonstrict hierarchical reinforcement learning for interactive systems and robots. TiiS **4**(3) (2014)
10. Sainath, T.N., Vinyals, O., Senior, A.W., Sak, H.: Convolutional, long short-term memory, fully connected deep neural networks. In: ICASSP. (2015)