Reward Shaping with Recurrent Neural Networks for Speeding up On-Line Policy Learning in Spoken Dialogue Systems

Pei-Hao Su, David Vandyke, Milica Gašić,
Nikola Mrkšić, Tsung-Hsien Wen and **Steve Young**
Department of Engineering, University of Cambridge, Cambridge, UK
{phs26, djv27, mg436, nm480, thw28, sjy}@cam.ac.uk

Abstract

Statistical spoken dialogue systems have the attractive property of being able to be optimised from data via interactions with real users. However in the reinforcement learning paradigm the dialogue manager (agent) often requires significant time to explore the state-action space to learn to behave in a desirable manner. This is a critical issue when the system is trained on-line with real users where learning costs are expensive. Reward shaping is one promising technique for addressing these concerns. Here we examine three recurrent neural network (RNN) approaches for providing reward shaping information in addition to the primary (task-orientated) environmental feedback. These RNNs are trained on returns from dialogues generated by a simulated user and attempt to diffuse the overall evaluation of the dialogue back down to the turn level to guide the agent towards good behaviour faster. In both simulated and real user scenarios these RNNs are shown to increase policy learning speed. Importantly, they do not require prior knowledge of the user's goal.

1 Introduction

Spoken dialogue systems (SDS) offer a natural way for people to interact with computers. With the ability to learn from data (interactions) statistical SDS can theoretically be created faster and with less man-hours than a comparable handcrafted rule based system. They have also been shown to perform better (Young et al., 2013). Central to this is the use of partially observable Markov decision processes (POMDP) to model dialogue, which inherently manage the uncertainty created by errors in speech recognition and semantic decoding (Williams and Young, 2007).

The dialogue manager is a core component of an SDS and largely determines the quality of interaction. Its behaviour is controlled by a *policy* which maps belief states to system actions (or distributions over sets of actions) and this policy is trained in a reinforcement learning framework (Sutton and Barto, 1999) where rewards are received from the environment, the most informative of which occurs only at the dialogues conclusion, indicating task success or failure.[1]

It is the sparseness of this environmental reward function which, by not providing any information at intermediate turns, requires exploration to traverse deeply many sub-optimal paths. This is a significant concern when training SDS on-line with real users where one wishes to minimise client exposure to sub-optimal system behaviour. In an effort to counter this problem, *reward shaping* (Ng et al., 1999) introduces domain knowledge to provide earlier informative feedback to the agent (additional to the environmental feedback) for the purpose of biasing exploration for discovering optimal behaviour quicker.[2] Reward shaping is briefly reviewed in Section 2.1.

In the context of SDS, Ferreira and Lefèvre (2015) have motivated the use of reward shaping via analogy to the 'social signals' naturally produced and interpreted throughout a human-human dialogue. This non-statistical reward shaping model used heuristic features for speeding up policy learning.

As an alternative, one may consider attempting to handcraft a finer grained environmental reward

[1] A uniform reward of -1 is common for all other, non-terminal turns, which promotes faster task completion.

[2] Learning algorithms are another central element in improving the speed of convergence during policy training. In particular the sample-efficiency of the learning algorithm can be the deciding factor in whether it can realistically be employed on-line. See e.g. the GP-SARSA (Gasic and Young, 2014) and Kalman temporal-difference (Daubigney et al., 2014) methods which bootstrap estimates of sparse value functions from minimal numbers of samples (dialogues).

Proceedings of the SIGDIAL 2015 Conference, pages 417–421,
Prague, Czech Republic, 2-4 September 2015.

function. For example, Asri et al. (2014) proposed diffusing expert ratings of dialogues to the state transition level to produce a richer reward function. Policy convergence may occur faster in this altered POMDP and dialogues generated by a task based simulated user may also alleviate the need for expert ratings. However, unlike reward shaping, modifying the environmental reward function also modifies the resulting optimal policy.

We recently proposed convolutional and recurrent neural network (RNN) approaches for determining dialogue success. This was used to provide a reinforcement signal for learning on-line from real users without requiring any prior knowledge of the user's task (Su et al., 2015). Here we extend the RNN approach by introducing new training constraints in order to combine the merits of the above three works: (1) diffusing dialogue level ratings down to the turn level to (2) add reward shaping information for faster policy learning, whilst (3) not requiring prior task knowledge which is simply unavailable on-line.

In Section 2 we briefly describe potential based reward shaping before introducing the RNNs we explore for producing reward shaping signals (basic RNN, long short-term memory (LSTM) and gated recurrent unit (GRU)). The features the RNNs use along with the training constraint and loss are also described. The experimental evaluation is then presented in Section 3. Firstly, the estimation accuracy of the RNNs is assessed. The benefit of using the RNN for reward shaping in both simulated and real user scenarios is then also demonstrated. Finally, conclusions are presented in Section 4.

2 RNNs for Reward Shaping

2.1 Reward Shaping

Reward shaping provides the system with an extra reward signal F in addition to environmental reward R, making the system learn from the composite signal $R + F$. The shaping reward F often encodes expert knowledge that complements the sparse signal R. Since the reward function defines the system's objective, changing it may result in a different task. When the task is modelled as a fully observable Markov decision process (MDP), Ng et al. (1999) defined formal requirements on the shaping reward as a difference of any potential function ϕ on consecutive states s and s' which preserves the optimality of policies. Based on this property, Eck et al. (2015) further extended it to POMDP by proof and empirical experiments:

$$F(b_t, a, b_{t+1}) = \gamma\phi(b_{t+1}) - \phi(b_t) \qquad (1)$$

where γ is the discount factor, b_t the belief state at turn t, and a the action leading b_t to b_{t+1}.

However determining an appropriate potential function for an SDS is non-trivial. Rather than hand-crafting the function with heuristic knowledge, we propose using an RNN to predict proper values as in the following.

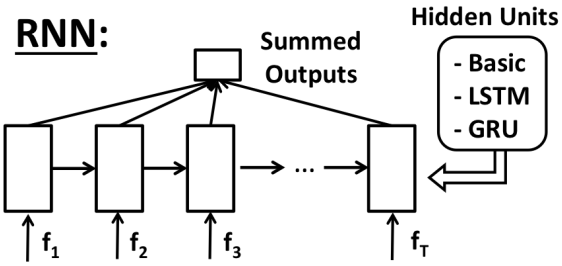

Figure 1: RNN with three types of hidden units: basic, LSTM and GRU. The feature vectors $\mathbf{f}_t$ extracted at turns $t = 1, \ldots, T$ are labelled $\mathbf{f}_t$.

2.2 Recurrent Neural Network Models

The RNN model is a subclass of neural network defined by the presence of feedback connections. The ability to succinctly retain history information makes it suitable for modelling sequential data. It has been successfully used for language modelling (Mikolov et al., 2011) and spoken language understanding (Mesnil et al., 2015).

However, Bengio et al. (1994) observed that basic RNNs suffer from vanishing/exploding gradient problems that limit their capability of modelling long context dependencies. To address this, long short-term memory (Hochreiter and Schmidhuber, 1997) and gated recurrent unit (Chung et al., 2014) RNNs have been proposed. In this paper, all three types of RNN (basic/LSTM/GRU) are compared.

2.3 Reward Shaping with RNN Prediction

The role of the RNN is to solve the regression problem of predicting the scalar return of each completed dialogue. At every turn t, input feature f_t are extracted from the belief/action pair and used to update the hidden layer h_t. From dialogues generated by a simulated user (Schatzmann and Young, 2009) supervised training pairs are created which consist of the turn level sequence of these feature vectors f_t along with the scalar dialogue

return as scored by an objective measure of task completion. Whilst the RNN models are trained on dialogue level supervised targets, we hypothesise that their subsequent turn level predictions can guide policy exploration via acting as informative reward shaping potentials.

To encourage good turn level predictions, all three RNN variants are trained to predict the dialogue return not with the final output of the network, but with the constraint that their scalar outputs from each turn t should sum to predict the return for the whole dialogue. This is shown in Figure 1. A mean-square-error (MSE) loss is used (see Appendix A). The trained RNNs are then used directly as the reward shaping potential function ϕ, using the RNN scalar output at each turn.

The feature inputs f_t for all RNNs consisted of the following sections: the real-valued belief state vector formed by concatenating the distributions over user discourse act, method and goal variables (Thomson and Young, 2010), one-hot encodings of the user and summary system actions, and the normalised turn number. This feature vector was extracted at every turn (system + user exchange).

3 Experiments

3.1 Experimental Setup

In all experiments the Cambridge restaurant domain was used, which consists of approximately 150 venues each having 6 attributes (slots) of which 3 can be used by the system to constrain the search and the remaining 3 are informable properties once a database entity has been found.

The shared core components of the SDS in all experiments were a domain independent ASR, a confusion network (CNet) semantic input decoder (Henderson et al., 2012), the BUDS (Thomson and Young, 2010) belief state tracker that factorises the dialogue state using a dynamic Bayesian network and a template based natural language generator. All policies were trained by GP-SARSA (Gasic and Young, 2014) and the summary action space contains 20 actions. Per turn reward was set to -1 and final reward 20 for task success else 0.

With this ontology, the size of the full feature vector was 147. The turn number was expressed as a percentage of the maximum number of allowed turns, here 30. The one-hot user dialogue act encoding was formed by taking only the most likely user act estimated by the CNet decoder.

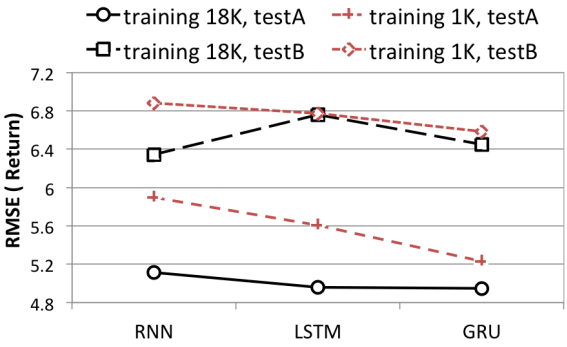

Figure 2: RMSE of return prediction by using RNN/LSTM/GRU, trained on 18K and 1K dialogues and tested on sets *testA* and *testB* (see text).

3.2 Neural Network Training

Here results of training the 3 RNNs on the simulated user dialogues are presented.[3] Two training sets were used consisting of 18K and 1K dialogues to verify the model robustness. In all cases a separate validation set consisting of 1K dialogues was used for controlling overfitting. Training and validation sets were approximately balanced regarding objective success/failure labels and collected at a 15% semantic error rate (SER). Prediction results are shown in Figure 2 on two test sets; *testA:* 1K dialogues, balanced regarding objective labels, at 15% SER and *testB:* containing 12K dialogues collected at SERs of $0, 15, 30$ and 45 as the data occurred (*i.e.* with no balancing regarding labels).

Root-MSE (RMSE) results of predicting the dialogue return are depicted in Figure 2. The models with LSTM and GRU units achieved a slight improvement in most cases over the basic RNN. Notice that the model with GRU even reached comparable results when trained with 1K training data compared to 18K. The results from the 1K training set indicate that the model can be developed from limited data. This enables datasets to be created by human annotation, avoiding the need for a simulated user. The results on set *testB* also show that the models can perform well in situations with varying error rates as would be encountered in real operating environments. Note that the dataset could also be created from human's annotation which avoids the need for a simulated user. We next examine the RNN-based reward shaping for policy training with a simulated user.

[3] All RNNs were implemented using the Theano library (Bergstra et al., 2010). In all cases the hidden layer contained 100 units with a sigmoid non-linearity and used stochastic gradient descent (per dialogue) for training.

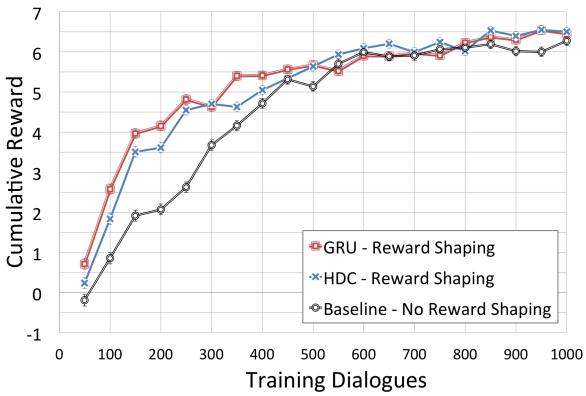

Figure 3: Policy training via simulated user with (GRU/HDC) and without (baseline) reward shaping. Standard errors are also shown.

Figure 4: Learning curves of reward with standard errors during on-line policy optimisation for the baseline (black) and proposed (green) systems.

3.3 Policy Learning with Simulated User

Since the aim of reward shaping is to enhance policy learning speed, we focus on the first 1000 training dialogues. Figure 2 shows that the GRU RNN attained slightly better performance than the other two RNN models, albeit with no statistical significance. Thus for clearer presentation of the policy training results we plot only the GRU results, using the model trained on 18K dialogues.

To show the effectiveness of using RNN with GRU for predicting reward shaping potentials, we compare it with the hand-crafted (HDC) method for reward shaping proposed by Ferreira and Lefèvre (2013) that requires prior knowledge of the user's task, and a baseline policy using only the environmental reward. Figure 3 shows the learning curve of the reward for the three systems. After every 50 training iterations each system was tested with 1000 dialogues and averaged over 10 policies. The simulated user's SER was set to 15%.

We see that reward shaping indeed provides the agent with more information, increasing the learning speed. Furthermore, our proposed RNN method further outperforms the hand-crafted system, whilst also being able to be applied on-line.

3.4 Policy Learning with Human Users

Based on the above results, the same GRU model was selected for training a policy on-line with humans. Two systems were trained with users recruited via Amazon Mechanical Turk: a baseline was trained with only the environmental reward, and another system was trained with an additional shaping reward predicted by the proposed GRU. Learning began from a random policy in all cases.

Figure 4 shows the on-line learning curve of the reward when training the systems with 400 dialogues. The moving average was calculated using a window of 100 dialogues and each result was averaged over three policies in order to reduce noise. It can be seen that by adding the RNN based shaping reward, the policy learnt quicker in the important initial phase of policy learning.

4 Conclusions

This paper has shown that RNN models can be trained to predict the dialogue return with a constraint such that subsequent turn level predictions act as good reward shaping signals that are effective for accelerating policy learning on-line with real users. As in many other applications, we found that gated RNNs such as LSTM and GRU perform a little better than basic RNNs.

In the work described here, the RNNs were trained using a simulated user and this simulator could have been used to bootstrap a policy for use with real users. However our supposition is that RNNs could be trained for reward prediction which are substantially domain independent and hence have wider applications via domain adaptation and extension (Gašić et al., 2015; Brys et al., 2015). Testing this supposition will be the subject of future work.

5 Acknowledgements

Pei-Hao Su is supported by Cambridge Trust and the Ministry of Education, Taiwan. David Vandyke and Tsung-Hsien Wen are supported by Toshiba Research Europe Ltd, Cambridge Research Lab.

References

Layla El Asri, Romain Laroche, and Olivier Pietquin. 2014. Task completion transfer learning for reward inference. In *Proc of MLIS*.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5(2):157–166.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. 2010. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference*.

Tim Brys, Anna Harutyunyan, Matthew E. Taylor, and Ann Nowé. 2015. Policy transfer using reward shaping. In *Proc of AAMAS*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Lucie Daubigney, Matthieu Geist, Senthilkumar Chandramohan, and Olivier Pietquin. 2014. A comprehensive reinforcement learning framework for dialogue management optimisation. *Journal of Selected Topics in Signal Processing*, 6(8).

Adam Eck, Leen-Kiat Soh, Sam Devlin, and Daniel Kudenko. 2015. Potential-based reward shaping for finite horizon online pomdp planning. *Autonomous Agents and Multi-Agent Systems*, pages 1–43.

Emmanuel Ferreira and Fabrice Lefèvre. 2013. Social signal and user adaptation in reinforcement learning-based dialogue management. In *Proc of MLIS*.

Emmanuel Ferreira and Fabrice Lefèvre. 2015. Reinforcement-learning based dialogue system for human-robot interactions with socially-inspired rewards. *Computer Speech & Language*, 34(1):256–274.

Milica Gasic and Stephanie Young. 2014. Gaussian processes for pomdp-based dialogue manager optimization. *TASLP*, 22(1):28–40.

Milica Gašić, Dongho Kim, Pirros Tsiakoulis, and Steve Young. 2015. Distributed dialogue policies for multi-domain statistical dialogue management. In *ICASSP*.

Matthew Henderson, Milica Gašić, Blaise Thomson, Pirros Tsiakoulis, Kai Yu, and Steve Young. 2012. Discriminative spoken language understanding using word confusion networks. In *IEEE SLT*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, et al. 2015. Using recurrent neural networks for slot filling in spoken language understanding. *TASLP*, 23(3):530–539.

Tomas Mikolov, Stefan Kombrink, Lukas Burget, Jan H Cernocky, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. In *ICASSP*.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*.

J. Schatzmann and S. Young. 2009. The hidden agenda user simulation model. *IEEE TALSP*, 17(4):733–747.

Pei-Hao Su, David Vandyke, Milica Gašić, Dongho Kim, Nikola Mrkšić, Tsung-Hsien Wen, and Steve Young. 2015. Learning from real users: Rating dialogue success with neural networks for reinforcement learning in spoken dialogue systems. In *Proc of Interspeech*.

Richard S. Sutton and Andrew G. Barto. 1999. *Reinforcement Learning: An Introduction*. MIT Press.

B. Thomson and S. Young. 2010. Bayesian update of dialogue state: A pomdp framework for spoken dialogue systems. *Computer Speech and Language*, 24:562–588.

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

Steve Young, Milica Gašic, Blaise Thomson, and Jason Williams. 2013. Pomdp-based statistical spoken dialogue systems: a review. In *Proc of the IEEE*, volume 99, pages 1–20.

A Training Constraint/Loss Function

For all RNN models the following MSE loss function is used on a per-dialogue basis:

$$\text{MSE} = \left(R - \sum_{t=1}^{T} r_t \right)^2 \tag{2}$$

where the current dialogue has T turns, R is the return and training target, and r_t is the scalar prediction output by the RNN model at each turn.