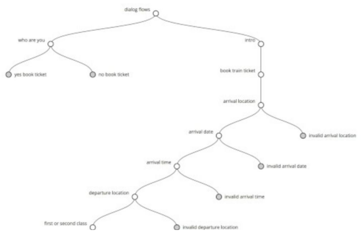
Deep Reinforcement Learning for Task-Oriented Dialogue

Nnamdi Iregbulem (nnamdi@stanford.edu) & Sahil Yakhmi (syakhmi@stanford.edu)

Problem

Most task-oriented chatbots are implemented as hand-coded **Finite State Machines (FSM)**, where user utterances matching pre-defined patterns transition the dialogue from one state to the next:



The open-source framework Rasa (rasa.com) has popularized a new architecture with flexible and learnable dialogue states and transitions.

However, this new architecture has a training data problem. Rasa requires training data for both NLU (intent recognition and parsing) and dialogue state management, but hand-annotating examples is a **time-consuming, expensive process**. Moreover, there is no simple way to leverage conversation data from chatbots deployed in the field.

Task

Prior research has attempted to combine reinforcement learning with traditional neural network-based natural language generation to varying degrees of success.

We demonstrate the viability of end-to-end Deep Reinforcement Learning for automatic and continuous learning for a chatbot agent.

We build a machine-learned action policy model to approximate the behavior of a deployed chatbot, leverage heuristics and conversational cues to extract rewards from unannotated conversation logs, and finally use Reinforcement Learning to improve performance over our baseline.

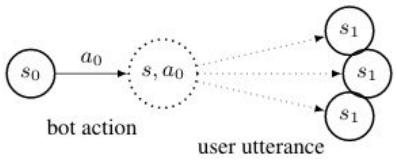
Data

Our primary dataset was obtained in partnership with Yalochat, Inc. and represented 30 days of real-world chatbot dialogue from the logs of the **Aeroméxico Facebook / WhatsApp chatbot**.

After cleaning, we were left with **14,647 conversations and 284,027 utterances**. Each user utterance was labeled with the state transition the production Aerobot performed after processing the input. After further cleaning of the labels, we were left with **98 labels**.

Approach

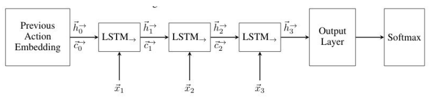
We modeled task-oriented dialogue as a **Markov Decision Process (MDP)** with infinite state space, allowing for the application of Reinforcement Learning (RL) techniques.



At each time-step, the bot chooses one of a finite set of pre-programmed actions to take based on the current state, which can be thought of as the entire dialogue up to that point. These actions might involve performing a computation, or interacting with a database, for example. After the bot performs an action, the dialogue will then transition non-deterministically to the next state, based on the probability distribution over possible user responses.

We chose an LSTM-based architecture with softmax output, initially taking as input the two most recent utterances and later augmenting the input with the previous bot action:

$$s_t = (a_{t-1}, b_{t-1,1}, b_{t-1,2}, \dots b_{t-1,n}, </s>, u_{t-1,1}, u_{t-1,2}, \dots u_{t-1,m})$$



Results

Baseline: Yalochat

Trained on the original Yalo state transition labels, our model achieved good fit to the data:

Dataset	dev	test
Accuracy	91.4%	91.4%
Weighted F1	0.913	0.910

Baseline + Improved State Representation: Yalochat

Building on the previous baseline, we made improvements to the model architecture and state representation. This became our primary baseline for later reinforcement learning experiments.

Dataset	dev	test
Accuracy	93.1%	92.7%
Weighted F1	0.930	0.924

Reinforcement Learning: Yalochat

Our model achieved high accuracy with respect to modeling Aerobot's behavior, but lower than the enhanced baseline system:

Dataset	dev	test	dev-baseline	test-baseline
Accuracy	91.8%	91.6%	93.1%	92.7%
Weighted F1	0.916	0.912	0.930	0.924

Analysis

In analyzing our RL system's performance, there was one question of primary interest---did our model learn the behaviors we were trying to reinforce?

Qualitatively, analyzing the model's performance on examples from the dev set shows that although reinforcement did not solve always solve the problem, **RL did boost the probability of the desired action on unseen data**. One example is presented below (correct action: (FS-Arrival}---flight search):

$s = ($ QnA-Question, "Escribe tu pregunta en un mensaje a continuación . Por ejemplo, : ¿ Puedo viajar con mi perro ?" "Cuanto vale el doblete de Oaxaca a Tijuana" $)$

a	FS-Arrival	QnA-QuestionScript	buenfin1a-yes	default
$Model_{RL}(s,a)$	0.918	0.078	0.001	0.001
$Model_{baseline}(s,a)$	0.000	0.989	0.005	0.002

Nevertheless, the RL process also created losses through overgeneralization, for example, interpreting nonsense input as a flight search.

$s = ($ default, "Todavía no puedo responder esa pregunta , <name> ." , "Jajajaa" $)$

a	FS-Arrival	QnA-Question	default	buenfin1a-yes
$Model_{RL}(s,a)$	0.517	0.312	0.117	0.002
$Model_{baseline}(s,a)$	0.000	0.001	0.729	0.004

Conclusion

In our experiments, we demonstrate the viability of approximating the behavior (action policy) of a hand-tuned, chatbot FSM with a deep neural architecture. These experiments represent a step in the direction of a **self-optimizing chat agent**, whose conversation flows do not have to be fully designed by a UX team and show the viability of **end-to-end learning of an action policy on raw text inputs** (without the use of an intermediate NLU knowledge representation), as well as the viability of reinforcement learning as a scalable learning technique within this context.

References

[1] Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. Deep reinforcement learning for dialogue generation. 2016.

[2] Pararth Shah, Dilek Hakkani-Tur, and Larry Heck. Interactive reinforcement learning for task-oriented dialogue management. 2016.

[3] Satinder Singh, Michael Kearns, Diane Litman, and Marilyn Walker. Reinforcement learning for spoken dialogue systems. 1999.