REINFORCEMENT LEARNING
AI Agent Assistant

Bexruz Nutfilloyev
Odilbek Umaraliev
Innopolis University, Innopolis, Tatarstan
b.nutfilloyev@innopolis.university
o.umaraliev@innopolis.university

I. Introduction

A chatbot is an interlocutor program that simulates human communication using text or voice. Chatbots help automate tasks by working according to a given algorithm. They conduct a dialogue with the user, fulfilling his requests, responding to requests or entertaining with their answers. The first programs simulating human communication appeared back in 1966. The virtual interlocutor Elisa quite convincingly parodied the dialogue with the therapist. With the growing popularity of messengers in the 2022s, chatbots have found a new life. Most work on platforms of popular messengers: Facebook Messenger, Telegram, Viber, VKontakte, Skype, Slack. Bots can work as separate applications or be integrated into the functionality of search engines.

Chatbots are used in areas such as e-commerce services, call centers, and the gaming industry. The use of chatbots for such purposes is usually limited to a narrow specialization, and they cannot be used for a wide range of communication with a person.

Different companies have different visions on how to classify chatbots. However, there are 2 types of classification: business classification of chatbot applications and classification of chatbot applications by technical type. There 3 types of chatbot applications of technical type “Based on business rules”, “Based on AI” and “Hybrid”. Our chatbot belongs to second type which is based on AI. In this project we have to build a chatbot which helps university students. The most important task in any university organization is to support students and answer their questions and try to solve their problems if students have any. In Innopolis University most of the time students want to an answer for questions about documents or any other type of questions like academic, administrative, extracurricular and etc. and they text to “Bot” which will be answered by the admin later. We wanted to make a bot that responds automatically and save students time. In this project we have built an “AI agent assistant bot” using Deep Reinforcement Learning and Natural Language Processing(NLP) techniques. The bot helps to answer for students questions instantly. The data is taken from “Frequently Asked Question” section in the university website. Since the data was not enough to get a satisfied result, we have used Reinforcement Learning and NLP techniques to improve the results and to learn from student questions. This is a convenient bot which learns from questions and answers in case if this question is already asked and answered by “StudentAffairs”, if user who asked the question does not satisfied for given respond, the user can “like” and “dislike” the answer and it is going to be a reward. By this reward the bot learns if this answer “acceptable” or not. Our model simulates dialogues between two virtual agents, using policy gradient methods to reward sequences that display three useful conversational properties: informativity, coherence, and ease of answering.

II. Problem Formulation

Our goal is to make a Telegram Chatbot which helps to momentarily get an answer for any question about Innopolis University.

III. Literature Review

There are a lot related works to this project. In this project we trained the data seq2seq model, which is classical model of Deep Neural Networks(DNNs). The seq2seq models is described in [1]. In this paper they present general end-to-end sequence learning approach. They used multilayered Long Short-Term Memory (LSTM) and Deep LSTM. The multilayered LSTM is used to map the input sequence to a vector of a fixed dimensionality and to decode the output sequence from the vector the Deep LSTM [2] is used. We have used 2 "Long Short-Term Memory"s which our seq2seq model is constructed. The main idea to handle variable-length input and output is to first encode the input sequence of words and then decode the output. To do that we have used the approach similar to the one described in paper [3] in which the encoder and decoder share the same weight.

After training the data we used Renforcement Learning technique called policy gradient to the chatbot in the future. This technique is used to generate more meaningfull answers to user based on already trained data. As we mentioned before in previous section that if the user could not get the answer he wanted, then the user can like and dislike the answer. These 2 actions is going to be a reward and helps to our telegram bot learning better and further improvement. The reward function is similar to the one described in a paper [4].

There is another case to respond for user question, in case if the user does not satisfied generated answer. In this case, the question will be sent to admin, who can sent more concrete answer to user. After getting respond from admin, this answer will be added to "MongoDB" database and improves the answer in the next time for a given question. But the one of the drawbacks of dialogue systems is that each module is trained individually, which presents several challenges. To solve this we used neural dialogue system which is described in [5]. In this paper [5] described how get an access for information in database to help users, to perform certain tasks and how dialogue system is interacted with database. Besides, also created a dialogue manager by using Reinforcement Learning. In case a dialogue system have noices. The dialogue manager helps to handle these noises.

The other related works focuses on building dialogue systems that are used in our project are [6] for User Agenda Modelling, [7] - used in Natural Language Generation (NLG) part, [8] used in Error Model Controller part. Those papers are all related to training and modeling the data using Neural Networks. In Reinforcement Learning part policy is represented as a Deep Q-network (DQN) [9], about critical points of RL training is described in [10].

IV. Methodology

A. Used Tools

1) Programming Languages: We used Python programming language which is commonly used in Data Science. In most of the related works the python with version 2.7 is used. In our project we used 3.9 version of python. There are multiple python tools and libraries is used to this project. Exmaples: for scraping data BeautifulSoup library, for building telegram bot Aiogram library and etc.

2) Data: The data is collected from Innopolis University website. From section "Frequently Asked Questions" (http://campuslife.innopolis.ru/faq), all the data was scrapped using Beautiful Soup which is a Python library for pulling data out of HTML and XML files. Unfortunately, the data was not quite enough to get more good results. There is 4 types of questions in this section. It includes academic, administrative, extracurricular (student clubs, innopoints and etc.) and the last one is a questions that are related to household (dorms, military services etc.). Totally we scrapped 43 questions with 43 answers.

3) Database: We used MongoDB database to store the data. MongoDB is NoSQL database which is used to build highly available and scalable internet applications.

B. Proposed Framework

1) Introduction and Problem Formulation: Deep Learning has an important role in society in various fields of activity and is at the heart of almost all the major computing breakthroughs of the last few years. And we all know that today the power of Deep Learning helps computers surpass human abilities. In the field of deep learning, neural models have been used in many dialogue systems as well. In papers [11] and [12] introduced network-based approach which takes the more similar answer from history and also authors applied an encoder and decoder models to train whole dialogue system. Using RL approach, which includes state tracking and policy learning, the first end-to-end dialogue system was presented in 2016 [13]. However, there were some drawbacks like responding only for *Yes/No* questions (inflexible question types) and robustness. In our project we improved and solved these drawbacks.

2) Dialogue system: The proposed schema of Dialogue system project divides into two parts. The first part is user simulator, and another part is a neural dialogue system. The framework is shown below in Figure 1

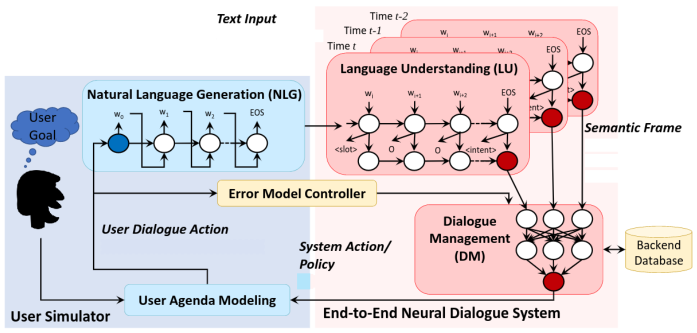

Figure 1. Framework of AI Agent Assistant project

To ensure that the user behaves consistently and goal-oriented, "User Agenda Modeling" is applied in the first part to regulate the dialogue exchange conditioned on the generated user goal. Natural language texts are produced by a Natural Language Generation(NLG) module in accordance with user dialogue actions. A neural dialogue system recognizes input sentences (voices or texts) and transforms them into semantic frames using Language Understanding(LU) modules. The Dialog Manager contains a state tracker and policy learner that accumulate semantics from utterances, record dialogue states during the conversation, and generate the next action on the system.

C. Neural Dialogue System

1) Language Understanding (LU): As part of its function, LU automatically classifies domains and intents of user queries and fills in slots to form a semantic frame.Slot tags are represented using the popular IOB (in-out-begin) format.

$$\vec{x} = w_1, ..., w_n$$

$$\vec{y} = s_1, ..., s_n, i_m$$

where $\vec{x}$ is the input word sequence, while $\vec{y}$ contains the associated slots and the sentence-level intent i_m. The Language Understanding component is implemented with Long short-term memory, which performs simultaneously both slot filling and intent prediction.

$$\vec{y} = LSTM(\vec{x})$$

Assuming a word sequence $\vec{x}$, LU maximizes the conditional probability of the slots and intent $\vec{y}$:

$$p(\vec{x}|\vec{y}) = (\prod_i^n) p(s_i|w_1, ..., w_i) p(i_m|\vec{y})$$

Backpropagation is used to train the weights of the LSTM model to maximize their conditional likelihood.It can be trained using all available dialogue actions and utterance pairs in the labeled dataset in a supervised fashion since the predicted tag set is a concatenated set of IOB-format slot tags and intent tags.

2) Dialogue Management (DM): It is a classic DM framework that includes two stages, dialogue state tracking and policy learning, in which the symbolic LU output passed into dialogue act(or semantic frame).

3) User Simulation: It is necessary to use a user simulator to automatically interact with the dialogue system for end-to-end training of the proposed neural dialogue systems.A user goal is generated by the user simulator in the task-completion dialogue setting. A user's goal is unknown to the agent, but the agent attempts to assist the user in achieving it.

4) User Agenda Modeling: In User Agenda the user state which is defined as s_u is factored into *A* (*A* is Agenda) and *G* (*G* is goal). *G* consists constraints *C* and reward *R*. Based on the current state $s_{u,t}$ and the last agent action $a_{m,t-1}$, the user simulator generates the next user action $a_{u,t}$ at each time step *t*.

5) Natural Language Generation (NLG): As a result of the user's dialogue actions, the NLG module generates natural language text.

6) Error Model Controller: In order to test the robustness of a policy based on semantic frames of user actions, an error model simulates noise coming from the user and agent communication as well as noise from the LU component.

D. End-to-End Reinforcement Learning

Finally, as part of the end-to-end reinforcement learning training process, each component of the neural network is fine-tuned to learn the interactive policy of our system. Deep Q-networks (DQNs) represent the policy. It takes state s_t from state tracker as input and outputs $Q(s_t, a; \theta)$ for all actions a

V. Results and Discussion

In this project we used end-to-end learning for dialog state tracking and management using deep reinforcement learning. We also build telegram bot to use this model in real world and this is originality of our project. As the bot is used, the more data will be collected from users and our bot will be improved based on user rewards and admin answers which are responded manually. One of the main advantages of our project is that the model can be improved in server automatically without any supervisor. The approaches that we used in this project are the latest researches and more improved solutions for dialog systems. Telegram bot is tested by real users. As a result the drawbacks that we mentioned in above sections is solved and performs better.

VI. Conclusion

Our goal was to make a Telegram Chatbot which helps to momentarily get an answer for any question about Innopolis University. It was really interesting to use for dialog systems with not only deep learning models but reinforcement learning as well. With the help of Reinforcement Learning the model is really improved and gets more exiting. We will share this bot with others to get feedbacks and solve drawbacks if any. We also uploaded this project to github. You can check it by clicking this link. As a feature work we are planning to build to this type of Telegram bots to make the students life more easier and upgrade our skills.

References

[1] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf

[2] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," 2016. [Online]. Available: https://arxiv.org/abs/1612.02695

[3] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, "Sequence to sequence - video to text," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[4] J. Li, W. Monroe, A. Ritter, M. Galley, J. Gao, and D. Jurafsky, "Deep reinforcement learning for dialogue generation," 2016. [Online]. Available: https://arxiv.org/abs/1606.01541

[5] X. Li, Y.-N. Chen, L. Li, J. Gao, and A. Celikyilmaz, "End-to-end task-completion neural dialogue systems," 2017. [Online]. Available: https://arxiv.org/abs/1703.01008

[6] J. Schatzmann and S. Young, "The hidden agenda user simulation model," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 4, pp. 733–747, 2009.

[7] T.-H. Wen, M. Gasic, N. Mrksic, P.-H. Su, D. Vandyke, and S. Young, "Semantically conditioned lstm-based natural language generation for spoken dialogue systems," 2015. [Online]. Available: https://arxiv.org/abs/1508.01745

[8] J. Schatzmann, B. Thomson, and S. Young, "Error simulation for training statistical dialogue systems," in *2007 IEEE Workshop on Automatic Speech Recognition Understanding (ASRU)*, 2007, pp. 526–531.

[9] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[10] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[11] T. H. Wen, M. Gasic, N. Mrksic, L. Rojas-Barahona, P.-H. Su, S. Ultes, D. Vandyke, and S. Young, "A network-based end-to-end trainable task-oriented dialogue system," 04 2016.

[12] A. Bordes, Y.-L. Boureau, and J. Weston, "Learning end-to-end goal-oriented dialog," *arXiv preprint arXiv:1605.07683*, 2016.

[13] T. Zhao and M. Eskenazi, "Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning," *arXiv preprint arXiv:1606.02560*, 2016.