A Survey on Reinforcement Learning for Dialogue Systems

Isabella Graßl
Chair of Intelligent Systems
University of Passau
Passau, Germany
grassl19@gw.uni-passau.de

***Abstract*—Dialogue systems are computer systems which communicate with humans using natural language. The goal is not just to imitate human communication but to learn from these interactions and improve the system's behaviour over time. Therefore, different machine learning approaches can be implemented with Reinforcement Learning being one of the most promising techniques to generate a contextually and semantically appropriate response. This paper outlines the current state-of-the-art methods and algorithms for integration of Reinforcement Learning techniques into dialogue systems.**

***Index Terms*—reinforcement learning, dialogue system, chatbot, conversational agent, human-computer-interaction**

I. Introduction

In recent years, a shift in human-computer-interaction towards a growing use of chat technology, especially so-called dialogue systems (DS), has been observed. DS are computer systems that communicate with humans using natural language, learn from these interactions and improve their behaviour over time.
DS become more and more important in society with humans interacting every day with personal assistants like Siri, Google Now, Cortana and Alexa – a fact which can also be illustrated by people's search behaviour on Google (Fig. 1). In March 2016 at Microsoft Build 2016, Microsoft CEO Satya Nadella introduced the term *conversation as a platform* (CaaP) to facilitate the creation of even more advanced personal assistants.

Fig. 1. The rise of DS illustrated by the search frequency of the term `chatbot` on Google Trends worldwide. The values indicate the search interest relative to the highest point on the graph in the specified time period, whereby the value 100 stands for the highest popularity of this search term.
Google Trends: https://trends.google.de/trends/explore?date=all&q=chatbot (accessed on 15.01.19)

In the context of DS, three major categories can be specified by their modality: spoken, text-based or multi-modal DS [1]. The intention of the user's interaction with the system is to achieve a certain goal using natural language whether it is in form of personal assistants or (social) chatbots. The goal of any DS is to figure out a satisfying dialogue strategy. This optimal dialogue strategy can be achieved manually in many ways. In any case, exploring, testing and evaluating strategies is time-consuming and their performance difficult to compare.
Recently, instead of actions based on static rules laid down by human developers, machine learning approaches use techniques such as Reinforcement Learning (RL) [2], so that the DS is able to learn strategies at runtime. This is related to the concept of Organic Computing (OC) [3] and the ideas behind autonomous systems, where – as a paradigm shift decisions made by a system are moved from design time to runtime.
In order to be able to use machine learning models in environments in which they can learn autonomous action-response events, learning processes are required which take into account the changing dynamics of the environment.
A popular example of the successful application of algorithms of this kind is the victory of Google's AI AlphaGo [4] over the worlds best human Go player. AlphaGo would not have been feasible with classical methods of supervised learning because – due to the intractable number of moves and scenarios – no model would have been able to describe the complexity of action-response relationships as a simple mapping between inputs and outputs. Instead, methods are needed that are able to independently respond to new circumstances of the environment, to anticipate possible future actions and to incorporate them into the current decision. The class of learning methods on which systems such as AlphaGo are based on is referred to as RL.
RL is the third large group of machine learning techniques besides Supervised and Unsupervised Learning. RL is a method based on the natural learning behaviour of humans. Human learning often occurs through simple exploration of the environment, especially in the early stages of learning.
Human actions within the framework of the learning problem are defined by a certain action-space. *Trial and Error* monitors and evaluates the effects of various actions on the environment. In response to our actions, we receive feedback from our environment, abstracted in the form of a reward or punishment. In many cases, the reward is paid in the form of

social acceptance, praise of other people but also by personal well-being or success. There is often a latency between action and reward where humans try to maximize the expected *total reward* over time through their actions rather than just generating immediate rewards [2].
RL refers to this psychological model, a form of goal-oriented learning where learning is derived by, from or during interactions with an external environment. Therefore, it is very well applicable to the field of DS – to create responses which are contextually and semantically appropriate. This paper outlines how the approach of RL is utilisable in the field of DS with its state-of-the-art methods and algorithms and discusses the future and limitations of this integration.
Besides human-computer-interaction applications such as DS or text summarization engines, the concept of RL is widely used in many different fields, e.g. in control tasks for robotics or helicopters [5], game playing like the already mentioned AlphaGo [4] or Chess [6] and also for consumer products like an autonomous vacuum cleaner.

II. Preliminaries

A. Reinforcement Learning

RL is based on the mathematical framework of the Markov Decision Process (MDP) and is a type of learning algorithm in which the system itself learns *what to do*, hence to learn which decisions on what actions to take in a certain situation or environment to maximise a numeric reward.
In general, a simple RL model consists of a set of states S, a set of actions A, a function of transition probabilities between states, a reward r_t and a discount factor γ to make a possible infinite reward sum finite. In principle, the process can be described as follows: an agent performs an action a_t in an environment for a given state s_t from the available action-space A, which results in a reaction of the environment in the form of a reward r_t. The reaction of the environment to the action of the agent in turn influences the agent's choice of the in the next state s_{t+1} (Fig. 2) [2].

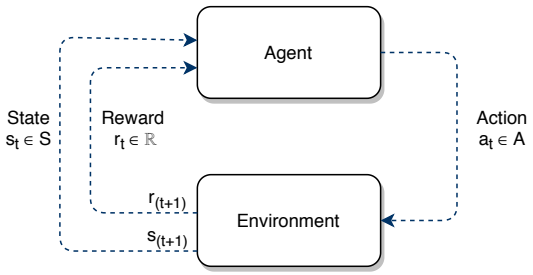

Fig. 2. General framework of RL with its agent-environment interaction where an agent takes actions in an environment which are rewarded and then fed back to the agent [2].

Over a certain number of iterations, the agent is able to approximate a relationship between its actions and the expected future benefits for a given state and thus to behave correspondingly optimally. In doing so, the agent has to strike a balance between using its previously acquired experience on the one hand and exploring new strategies to increase rewards on the other hand. This is called the exploration-exploitation dilemma [2].
RL also contains important sub-elements: a policy, meaning a mapping from environment states to actions making up the behaviour of the agent, a reward function that defines a specific goal or what is *good* or *bad* behaviour, a value function of what is a *good* action due to the expected reward in long-term performance and a model that outlines the impact of the actions [2].
There are three different types of RL: passive, active and deep RL. In passive RL, an agent executes a fixed policy defined at design time given by a human developer. In contrast, a system using active RL [7] updates its policy as it learns with the goal to learn an optimal policy. Active RL is a combination of the advantages of two basic approaches: active learning has been combined with RL for determining the sensitivity of the optimal policy to changes in state transitions and rewards, but not in the actual environment due to time-extensity and high risk. Active RL is used to explore regions of the state-action space where the optimal policy is maximally uncertain [7]. Deep RL is an extension of active RL based on neural networks and is most commonly used.

B. Dialogue Systems

There are many different architectures for different DS, all based on the same set of main components and phases forming a certain input-output-flow (Fig. 3 on the following page).
This cycle begins with so-called input modules which recognize user input and convert it to a textual representation, i.e. a string of words, if necessary. In case of a speech-based DS, speech recognition is performed Speech-to-Text technologies based on phonetics and phonology [1].
As a next step, the intention of the user is transferred into structured data, which is mostly done with Natural Language Understanding (NLU) frameworks. As the telling name suggests, the task is to figure out the intention of the user. Therefore, the interpretation needs to include several aspects such as a syntactic, discourse, semantic, and pragmatic analysis [8]. This includes the investigation of grammatical relationships among the words by parsing the sentence as well as its meaning. If there is more than one sentence, the relationship between those is determined via co-reference resolution such as anaphora or cataphora in discourse analysis.
This interpretation is provided to the Dialogue Manager (DM). It is the main component of a DS and responsible for selecting the most appropriate action as a response to a statement. This includes also the maintenance of dialogue history and adopting dialogue strategies. As the most trivial strategy, there is only one dialogue state, corresponding to the goal of answering the next question. The new dialogue state is a function of the current state, the user's statement,

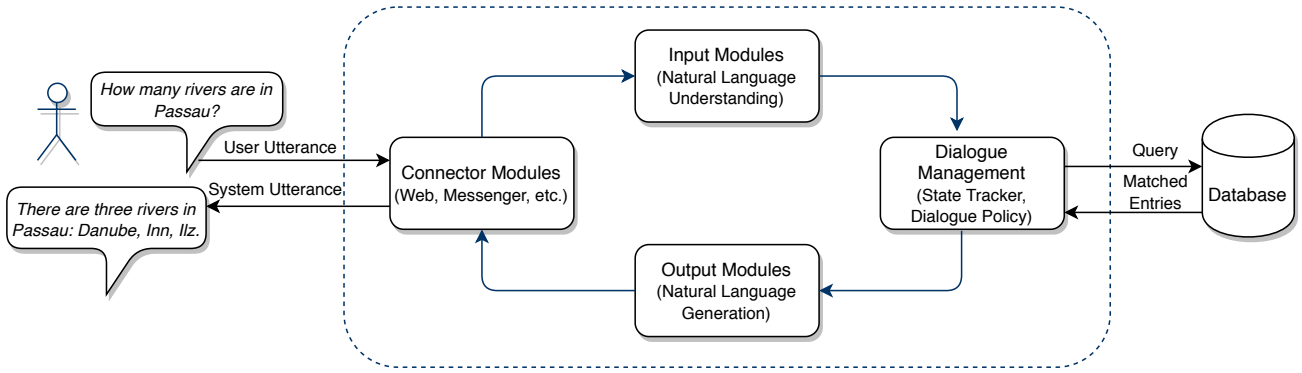

Fig. 3. Simplified general architecture of a DS, combining the most common properties from the different architectures of DS. The dialogue can be modelled as an RL-problem where the DS is the RL agent and the user is the environment; based on [1] [8] [9].

and the subject of the conversation. The DM is split into the State Tracker (ST) and the policy for the agent, which can be represented by a neural network in case of deep RL [1].
In most cases, the DM must access external knowledge in the form of domain-specific content (e.g. flight plans or movie show times). As a consequence, the DM needs to interact with some kind of external knowledge source such as a database. The queries must be converted to the format of the external system.
Eventually, a response is generated within the output modules and their Natural Language Generator (NLG) [1]. This constructed message can be either in form of a simple communication to the user such as spoken or written text, showing web pages or non-verbal responses, e.g. updating the day planner. If the system is not text-based, Text-to-Speech technologies such as Natural Language Synthesis are used to generate the output. Pre-recorded speech, such as *Hello, how may I help you?*, can be used to facilitate the start of a conversation.

III. Integration of RL into DS

The RL process for DS can be described as an agent studying how the user has replied and tracking positive signals such as *Thank you*. It tries to identify behaviour patterns and learn from interactions from different persons. Based on the analysis, policies are framed which, in turn, are the basis for situation-dependent behaviour. The agent then collects experiences to reframe its policies. Deep RL methods are used for developing more accurate policies.
The MDP models a system's interaction with human users while RL is used to optimize the systems performance. The amount of training data is limited by the fact that a human has to interact with the system to obtain it. In DS, RL is used to optimize the design of a dialogue management policy (Fig. 4).

The queries of the dialogue manager are modelled as states and answers of the database are modelled as possible actions. For some states, the proper action to take may be clear as for instance, greeting the user in the start state as already mentioned.

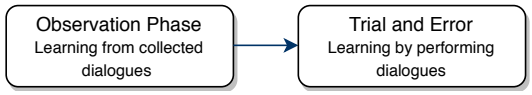

Fig. 4. Training of the DM with RL to reframe and optimize its policy; own representation.

The reward function depends on the feedback of the dialogue manager based on the user choices. This approach allows the system to evolve in an uncertain environment and to collect rewards. Intrinsically motivated rewards could also capture aspects of the user's emotional satisfaction in human-computer-interactions. The appropriate measure for the success of a DS can range from user satisfaction to task completion or sales figures in commercial applications to the number of times users have to interrupt the system or returned sentences from the system like *Sorry, I do not understand*. The dialogue ST processes the semantic frame, ergo the input data and its value, and the history of the current conversation into a state representation that can be used by the agent's policy. This state is then fed as input into the policy of the agent, in deep RL by a neural network, to produce an action which is passed to the output modules (cf. again Fig. 3).

IV. Taxonomy of RL for DS

In general, DS bots can be classified into four major types: social chatbots, infobots, task completion bots (task-oriented or goal-oriented) and personal assistant bots [1].
There are three generations of dialogue technology since the early '60s. The first generation focused on grammatical rules and is based on human experts. The second generation used data to learn statistical parameters in DS [5]. Since 2014, the third generation uses neural models in addition to such statistical parameters.
Based on the common types of DS and the historical background of dialogue technology, the results of this survey are structured by a classification system illustrated in Fig. 5. [1]

[1]Literature databases were used to collect and review publications with the results being based on Google Scholar which is aware of most influential research papers [10].

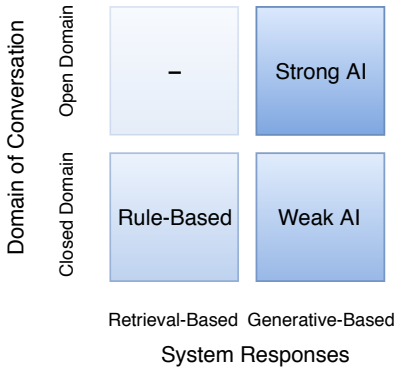

Fig. 5. A classification scheme of DS categorized by ordered pairs of the conversation ability on the ordinate and the response mechanism on the abscissa whereby DS can be based on rules or artificial intelligence (AI). A retrieval-based DS in an open domain is not possible whereas a DS which retrieves responses for a predefined topic domain is the comparatively simplest form of a DS. If the DS is able to create new responses in a closed domain, it is regarded as a weak AI. In contrast, a generative-based DS without limitations of a conversational domain would have *true intelligence*.
Based on Kojouharov (2016): https://chatbotslife.com/@kojouharov (accessed on 15.01.19)

A. Closed-Domain and Retrieval-Based DS

In the first generation, rule-based or corpus-based DS were implemented and are still used for common goal-oriented chatbots in a specific conversational domain (quadrant III in Fig. 5). The purpose of goal-oriented or task-oriented dialogue agents, i.e. agents interacting in a closed-domain, is to solve a single problem for a user such as making a reservation or booking.
Most of the task-oriented systems use so-called *slot-filling methods* to capture the user request in a domain-specific conversation. Concrete, for making a reservation in a restaurant, the intent performs an action in response to natural language user input, like `RestaurantBooking` and utterances as spoken or written phrases which convey the user's intent, like *I want to make a reservation in a restaurant*. From an information processing perspective, slots are input data required to fulfil this concrete intent, for instance, the value of location, price range or opening times. After all slots are filled, the fulfilment mechanism for the user's intent takes action, for example in the response: *Your reservation for the restaurant was successfully made*. Each human-machine-interaction contains slot-value pairs which are the so-called semantic frames.
This traditional approach has proven reliable, but cannot be adapted to other domains – even if they share common intents or slots.
One of the most famous chatbots and probably the first program which faced the Turing Test[2] was ELIZA [11] in 1966. As ELIZA is a rule-based system for a closed-domain, it could not have any meaningful conversations with humans except for single predefined task – it had generic responses to utterances which did not fit into any rules.
ALICE [3] is a popular free chatbot developed by Richard Wallace in 1995. It is influenced by ELIZA and won the Loebner prize in January 2000.
One of the most successful approaches for a task-oriented system was introduced by Young et al. (2013) [12] where they define dialogue as a Partially Observable Markov Decision Process (POMDP).
In contrast to the before mentioned MDP, the states in a POMDP are not completely observable, but as in MDP, the state transitions can be controlled. In the centre of the architecture of such DS are two stochastic models: a dialogue model and a policy model. The dialogue model has a transition probability $p(s_t \mid s_{t-1}, a_{t-1})$ and an observation probability $p(o_t \mid s_t)$, where s_t is the state of the dialogue at time t, a_t is the action taken at time t, and o_t is the observation at time t.
The policy model determines which action to take at each turn. As the dialogue progresses, a reward is given at each step to reflect the desired characteristics of the DS. The dialogue model M and policy model P can be optimized using RL from these rewards either through interaction with users or from a corpus of dialogues (Fig. 6 on the previous page) [12].

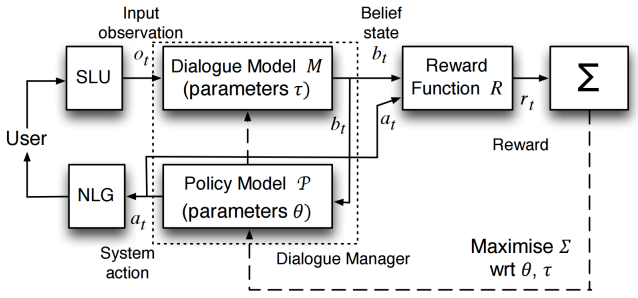

Fig. 6. The dialogue model and policy model form the centre of the task-oriented system with belief state tracking and RL. In contrast to the MDP, the input utterance is regarded as an observation of the underlying user intent which is hidden. Instead of trying to estimate the hidden dialogue state, the system response is directly given from the distribution over all possible dialogue states [12].

The launch of Siri in 2010 as an intelligent virtual assistant pioneered for other personal assistants like Google Now (2012), Cortana (2015) and Alexa (2015). All of these assistants can help answer questions, partly based on the web and want to extend the capability of not just focusing on one narrow task.
Therefore, the retrieval-based methods must be replaced with approaches which are capable of creating new responses in addition to copying or mapping user utterances to a system response by rules or sequence-to-sequence methods.

[2]Alan Turing introduced his Turing Test in 1950 to survey a machine's ability to be seen as *human* in an interaction with another human being. As of today, the Turing Test might not be exactly a proof of consciousness of a program but is a reliable indicator, especially in the field of human interaction.

[3]Wallace (1995): http://www.alicebot.org (accessed on 15.01.19)

B. Closed-Domain and Generative-Based DS

As IBM's Watson[4] was developed in 2006 specifically to compete with the human champions in the game *Jeopardy!* in 2011, it was a rule-based system in a closed-domain. Since then, IBM Watson offers services to build chatbots for various domains. These are not just able to process large amounts of data and to fill slots like retrieval-based DS but also to generate new responses for certain tasks and can, therefore, be categorized as a weak AI in the conversational context (quadrant IV in Fig. 5 on the preceding page).[5]

Li et al. (2017) [13] introduced a task-oriented DS with its parameters trained by using supervised learning as well as RL. The system is specified for the topic area of a movie booking. As they used Deep Q-Networks (DQN), a high amount of data must be available which can prove challenge.

Similary, Williams and Zweig (2016) [14] introduced an approach using a combination of supervised learning and RL, but contrary to other approaches, they are able to use both for optimizing the policy function. They used a long short-term memory (LSTM) neural network to remember past observations arbitrarily long and enabling them to reduce the costs of designing an appropriate state space.

Most of those deployed DS use manual features for the state and action-space representation and require either a huge amount of annotated domain-specific data or people willing to interact with an unfinished system [15]. This not only makes it expensive and time-consuming to deploy a real DS but also limits its usage to a narrow domain.

C. Transfer to Open-Domain DS

Towards more advanced DS: If the system can transfer knowledge from one domain to another, it can be called a strong or general AI. Microsoft is aiming fur such system with its Chinese bot Xiaoice[6] which interacts with people on messaging platforms in Asia and America (quadrant I in Fig. 5 on page 4). So far, most of the common systems like Alexa and Siri just unify different domains to seem like a strong AI.

Conversational open-domain systems are systems which are not limited to a certain conversational domain. In the context of social chatbots, so-called chitchat-bots or chatterbots are often mentioned which attempt to provide full-dialogue. In general, generative methods produce responses one at a time without considering their long-term effect.

In their work, Peng et al. (2017) [16] created a DS considering more than a single-domain using deep RL and hierarchical task decomposition to also reduce costs of state-action space. They applied their system in the context of booking a hotel and flight while also considering time for commuting with a rental car using a two-level dialogue policy. As more complex (sub-)tasks require multiple levels of hierarchy, they can extend their approach, but this would be inefficient and time-consuming in return.

The approach of Ilievski et al. (2018) [17] also addresses the single-domain problem as they introduced a goal-oriented DS using a transfer learning method based on earlier work of [13] and [18][7]. As both approaches implemented bots independently for their respective domain, [17] are utilising the similarities between a source and target domain and transferring the knowledge from one neural network to another. As illustrated in Fig. 7, two domains can include the same type of information and therefore, there is no need for learning this particular information twice. This transfer learning technique can also be applied if a third domain, for example, the domain `Tourism`, is added which shares all information from the source domain and some additional information like type of accommodation [17].

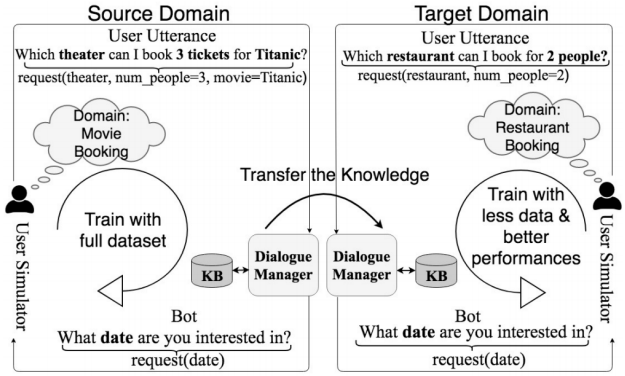

Fig. 7. Model of the DS where a knowledge transfer can be realised due to a domain overlap of the source domain `MovieBooking` and the target domain `RestaurantBooking`, both sharing the same information `date` [17].

A further step towards open domain chatbots is from a research group at the Montreal Institute of Learning Algorithms with MILABOT which successfully reached the semi-finals of the Alexa Prize competition: Serban et al. (2017) [19] implemented MILABOT, a chatbot using an entirety of 22 individual models including sequence-to-sequence methods to create responses and then choosing the most appropriate response with the help of deep RL. On the downside, they had also issues dealing with speech recognition errors which can have an impact on the user's experience with the system.

[4]The DeepQA Research Team (2006): https://www.ibm.com/watson (accessed on 15.01.19)

[5]The term `strong AI` is referring to John Searle's thought experiment of the Chinese room in 1980 and can also be regarded as artificial general intelligence (AGI). As a constructive criticism of the Turing Test, Searle discusses whether a machine is capable of understanding (a conversation) or just simulating.

[6]Microsoft (2014): http://www.msxiaoice.com (accessed on 15.01.19)

[7]In analogy to [13], Wen et al. (2016) [18] presented a goal-oriented DS for restaurant reservations by using a neural network-based model based on a Wizard-of-Oz paradigm, in which generally a human secretly tells the agent which actions to take, instead of RL.

V. Discussion

End-to-end statistical conversational models are already satisfying, but open-domain (chat)bots are still at an early state where one of the main problems remains the access to external databases for domain-specific knowledge and the transfer to other domains, as the approach of [17] showed.
Besides, one of the key problems is a lack of understanding of natural language by the system. Especially in spoken DS, ill-formed utterances like dialect which are not trivial to take care of at runtime and may need consideration at design time. One of the more significant problems is the co-reference resolution like anaphora or cataphora.
In contrast, the syntax level of the responses of the generative-based models can be quite low as they create their answers and not just reply with one of the pre-defined answers from a set of possible answers. As a consequence, they need more training compared to the rule-based approach which can be problematic, but might handle complex and unseen utterances in return.
In addition, a fundamental problem of the approach of RL in practice is to extract reliable feedback considering the reward from the user [15]. In many application fields, the reward is easy to determine such as with games where the system get `+1` if it wins and `0` or `-1` otherwise [4] [6].
Regarding a simple feedback model by just asking the user if the application was helpful, a `Yes` can merely indicate politeness whereas a `No` can be the result of disappointment or frustration at the end of the user. Thereby the user's perception of the system can vary significantly from the task completion success.

VI. Conclusion and Future Work

This paper discussed how the approach of RL can be used in the field of DS and its state-of-the-art methods. It showed that RL is a promising approach to generate intuitive responses which are contextually and semantically appropriate.
RL, as well as (chat)bots, attracted much attention not just in the area of computer science but also in society in general. However, the conception and implementation of DS using RL faces many difficulties as discussed before.
DS offer new ways of interaction between computer and humans and can be integrated into any domains and processes such as education, healthcare, organisation and management, business or entertainment. RL is a promising tool for solving automated natural language processing across those domains and transferring it to other domains as well as being interlingually.
However, further research has to be done towards systems which actually *understand* semantics. This implies also the integration of psychological as well as linguistic knowledge into the designing and implementation process, forming an interdisciplinary approach.
One step towards this future of DS was taken by Google: they announced in early January 2019 at the CES 2019 that Google Assistant is on 1 billion devices, can perform over a million of actions and is conversant in 30 languages.

References

[1] Jurafsky, D. and Martin, J. H. (2018): Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Prentice-Hall. Draft of September 23, 2018. [online] https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf (accessed on 15.01.19)
[2] Sutton, R. S., & Barto, A. G. (1998): Introduction to reinforcement learning (Vol. 135). Cambridge: MIT press.
[3] Müller-Schloer, C., & Tomforde, S. (2017): Organic Computing–Technical Systems for Survival in the Real World. Springer International Publishing.
[4] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M. (2016): Mastering the game of go with deep neural networks and tree search. Nature, 529(7587),484-489.
[5] Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996): Reinforcement learning: A survey. Journal of artificial intelligence research, 4, 237-285.
[6] Lai, M. (2015): Giraffe: Using deep reinforcement learning to play chess. arXiv preprint arXiv:1509.01549.
[7] Epshteyn, A., Vogel, A., & DeJong, G. (2008): Active reinforcement learning. In Proceedings of the 25th international conference on Machine learning, 296-303. ACM.
[8] Singh, M. (2004): Practical Handbook of Internet Computing. Chapman and Hall/CRC.
[9] Petraityt, J. (2018): Deprecating the state machine: building conversational AI with the Rasa stack, PyData Berlin 2018.
[10] Martin-Martin, A. (2017): Can we use Google Scholar to identify highly-cited documents?
[11] Weizenbaum, J. (1983): ELIZA - A Computer Program for the Study of Natural Language Communication Between Man and Machine, Communications of the ACM, vol. 26, no. 1, 23-28.
[12] Young, S., Gai, M., Thomson, B., & Williams, J. D. (2013): Pomdp-based statistical spoken dialogue systems: A review. Proceedings of the IEEE, 101(5), 1160-1179.
[13] Li, J., Monroe, W., Ritter, A., Galley, M., Gao, J., & Jurafsky, D. (2017): Deep reinforcement learning for dialogue generation. arXiv preprint arXiv:1606.01541.
[14] Williams, J. D., & Zweig, G. (2016): End-to-end lstm-based dialogue control optimized with supervised and reinforcement learning. arXiv preprint arXiv:1606.01269.
[15] Su, P. H., Vandyke, D., Gasic, M., Kim, D., Mrksic, N., Wen, T. H., & Young, S. (2015): Learning from real users: Rating dialogue success with neural networks for reinforcement learning in spoken dialogue systems. arXiv preprint arXiv:1508.03386.
[16] Peng, B., Li, X., Li, L., Gao, J., Celikyilmaz, A., Lee, S., & Wong, K. F. (2017): Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning. arXiv preprint arXiv:1704.03084.
[17] Ilievski, I., Musat, C., Hossmann, A., & Baeriswyl, M. (2018): Goal-Oriented Chatbot Dialog Management Bootstrapping with Transfer Learning. arXiv preprint arXiv:1802.00500.
[18] Wen, T., Vandyke, D., Mrksic, N., Gasic, M., Rojas-Barahona, L., Su, P., Ultes, S. & Young, S. (2016): A network-based end-to-end trainable task-oriented dialogue system.arXiv preprint arXiv:1604.04562.
[19] Serban, I. V., Sankar, C., Germain, M., Zhang, S., Lin, Z., Subramanian, S. & Rajeswar, S. (2018): A Deep Reinforcement Learning Chatbot (Short Version). arXiv preprint arXiv:1801.06700.