Artificial Intelligence Review
https://doi.org/10.1007/s10462-022-10205-5

Survey on reinforcement learning for language processing

Víctor Uc-Cetina[1,3] · Nicolás Navarro-Guerrero[2] · Anabel Martin-Gonzalez[1] · Cornelius Weber[3] · Stefan Wermter[3]

Abstract
In recent years some researchers have explored the use of reinforcement learning (RL) algorithms as key components in the solution of various natural language processing (NLP) tasks. For instance, some of these algorithms leveraging deep neural learning have found their way into conversational systems. This paper reviews the state of the art of RL methods for their possible use for different problems of NLP, focusing primarily on conversational systems, mainly due to their growing relevance. We provide detailed descriptions of the problems as well as discussions of why RL is well-suited to solve them. Also, we analyze the advantages and limitations of these methods. Finally, we elaborate on promising research directions in NLP that might benefit from RL.

Keywords Reinforcement learning · Natural language processing · Conversational systems · Parsing · Translation · Text generation

1 Introduction

Machine learning algorithms have been very successful to solve problems in the natural language processing (NLP) domain for many years, especially supervised and unsupervised methods. However, this is not the case with reinforcement learning (RL), which is

✉ Víctor Uc-Cetina
uccetina@correo.uady.mx

Nicolás Navarro-Guerrero
nicolas.navarro@dfki.de

Anabel Martin-Gonzalez
amarting@correo.uady.mx

Cornelius Weber
weber@informatik.uni-hamburg.de

Stefan Wermter
wermter@informatik.uni-hamburg.de

[1] Universidad Autónoma de Yucatán, Anillo Periférico Norte, 97119 Mérida, Mexico

[2] Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Robert-Hooke-Straße 1, 28359 Bremen, Germany

[3] Universität Hamburg, Vogt-Kölln-Str. 30, 22527 Hamburg, Germany

Published online: 03 June 2022

somewhat surprising since in other domains, RL methods have experienced an increased level of success with some impressive results, for instance in board games such as AlphaGo Zero (Silver et al. 2017). Yet, deep RL (DRL) for natural language processing is still in its infancy when compared to supervised learning (LeCun et al. 2015). Thus, the main goal of this article is to provide a review of applications of reinforcement learning to NLP. Moreover, we present an analysis of the underlying structure of the problems that make them viable to be treated entirely or partially as RL problems, intended as an aid to newcomers to the field. We also analyze some existing research gaps and provide a list of promising research directions in which natural language systems might benefit from RL algorithms.

1.1 Reinforcement learning

RL is a term commonly used to refer to a family of algorithms designed to solve problems in which a sequence of decisions is needed. RL has also been defined more as a kind of learning problem than as a group of algorithms used to solve such problems (Sutton and Barto 2018). It is important to mention that RL is a very different kind of learning than the ones studied in supervised and unsupervised methods. This kind of learning requires the learning system, also known as agent, to discover by itself through the interaction with its environment, which sequence of actions is the best to accomplish its goal.

There are three major groups of reinforcement methods, namely, dynamic programming, Monte Carlo methods, and temporal difference methods. Dynamic programming methods estimate state or state–action values by making estimates from other estimates. This iteratively intertwines policy evaluation and policy improvement updates taking advantage of a model of the environment which is used to calculate rewards. Policy evaluation consists of updating the current version of the value function based on the current policy. Policy improvement consists of greedifying the policy function based on the current value function. Depending on the algorithm and its implementation it might require exhaustive sweeping of the entire state space or not. Monte Carlo methods learn from complete sample returns, instead of immediate rewards. Unlike dynamic programming, Monte Carlo methods only consider one transition path at a time, the path generated with a sample. In other words, they do not bootstrap from successor states' values. Therefore, these kinds of methods are more useful when we do not have a model of the environment, the so-called dynamics of the environment. Temporal difference methods do not need a model of the environment since they can learn from experience, which can be generated from interactions with the environment. These methods possess the best of dynamic programming and the best of Monte Carlo. From dynamic programming they inherit the bootstrapping, from Monte Carlo methods they inherit the sampling. As a result of this combination of characteristics, temporal difference methods have been the most widely used. All these methods pose the decision-making problem as a Markov decision process (MDP). An MDP is a mathematical method used to solve decision-making in sequence and considers as the minimum existing elements a set of states S, a set of actions A, a transition function T, and a reward function R. Given an MDP (S, A, T, R), we need to find an optimal policy function π, which represents the solution of our sequence decision problem. The aim of a RL system, or so-called agent, is to maximize some cumulative reward $r \in R$ through a sequence of actions. Each pair of state s and action a creates a transition tuple (s, a, r, s'), with s' being the resulting state. Depending on the algorithm being used and on the particular settings of our problem, the policy π will be estimated differently.

A policy π defines the behavior of the agent at any given moment. In other words, a policy is a mapping from the set of states S perceived from the environment to a set of actions A that should be executed in those states. In some cases, the policy can be stored as a lookup table, and in other cases it is stored as a function approximator, such as a neural network. The latter is imperative when we have a large number of states. The policy is the most important mathematical function learned by the reinforcement learning agent, in the sense that it is all the agent needs to control its behavior once the learning process has concluded. In general, a policy can be stochastic and we formally define it as $\pi : S \rightarrow A$.

The goal in a RL problem is specified by the reward function R. This function maps each state or pair of state-action perceived in the environment to one real number $r \in \Re$ called reward. This reward indicates how good or bad a given state is. As we mentioned before, the goal of an agent is to maximize the total amount of rewards that it gets in the long run, during its interaction with the environment. The reward function cannot be modified by the agent, however, it can serve as a basis for modifying the policy function. For example, if the action selected by the current policy is followed by a low reward, then the policy can be updated in such a way that in the future it indicates a different action when the agent encounters the same situation. In general, the reward function can also be a stochastic function and it is formally defined as $R : S \rightarrow \Re$, or $R : S \times A \rightarrow \Re$.

A value function indicates which actions are good in the long run. The value of a state is basically an estimation of the total amount of rewards that the agent can expect to accumulate in the future, if it starts its path from that state using its current policy. We should not confuse the value function with the reward function. The rewards are given directly by the environment while the values of the states are estimated by the agent, from its interaction with the environment. Many RL methods estimate the policy function from the value function. When the value function is a mapping from states to real numbers, it is denoted by the letter V. When the mapping is from pairs of state-action to real numbers, it is denoted by Q. Formally, we can define the value function as $V : S \rightarrow \Re$ or $Q : S \times A \rightarrow \Re$.

In the case of model-based RL, the agent also has access to a model of the transition function T of the environment, which may be learnt from experience. For example, given a state and an action, the model could predict the next resulting state and reward. Such world models are used for planning, this is, a way to make decisions about the next actions to be performed, without the need to experience possible situations. In the case of model-free RL, when a model of the environment is missing, we have to solve the RL problem without planning and that means that a significant amount of experimentation with the environment will be needed.

One of the most popular RL algorithms is the Q-learning algorithm (Watkins 1989). As its name suggests, it works by estimating a state-action value function Q. The algorithm does not rely on a model of the transition function T, and therefore it has to interact with the environment iteratively. It follows one policy function for exploring the environment and a second greedy policy for updating its estimations of the values of pairs of states and actions that it happens to visit during the learning process. This kind of learning is called off-policy learning. The algorithm uses the following rule for updating the Q values:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right].$$

In this learning rule, α is a parameter defined experimentally and it is known as the learning rate. It takes values in the interval (0, 1). Moreover, r is the reward signal, γ is known as the discount parameter and it also takes values in the interval (0, 1), and finally s' and a'

denote the next state and the next action to be visited and executed during the next interaction with the environment.

SARSA is an on-policy learning algorithm, meaning that instead of using two policies, one for behavior and one for learning, this algorithm uses only one policy. The same policy that is used to explore the environment is the same policy used in the update rule (Sutton and Barto 2018). The update rule of the SARSA is the following:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma Q(s',a') - Q(s,a)].$$

A very important result in recent years was the development of the deep Q-network (DQN, Mnih et al. 2015), in which a convolutional neural network is trained with a variant of Q-learning. This algorithm, originally designed to learn to play several Atari 2600 games at a superhuman level, is now being applied to other learning tasks. Another algorithm, AlphaGo Zero (Silver et al. 2016), learned to play Go and actually defeated the human world champion in 2016. This algorithm uses a deep neural network, a search algorithm and RL rules. The successor model MuZero (Schrittwieser et al. 2020) learns a representation of state, a dynamics and a reward prediction function to maximize future rewards via tree search-based planning, achieving more successful game play without prior knowledge of the game rules.

DRL is an extension of the classical RL methods to leverage the representational power of deep models. More specifically, deep neural networks allow RL algorithms to approximate and store highly complex value functions, state-action functions, or policy functions. For instance, a $Q(s, a)$ function can be represented as a convolutional neural network or a recurrent one. Similarly to what happened in other domains such as computer vision, deep models are also playing a decisive role in the advancement of RL research, especially in MDPs with very large state and action spaces. In fact, reinforcement learning and deep neural networks have stayed recently at the center of attention of many researchers who have studied and applied them to solve different problems, including problems in NLP, as we will discuss below.

1.2 Natural language processing and RL

In NLP, one of the main goals is the development of computer programs capable of communicating with humans through the use of natural language. In some applications, such as machine translation (MT), these programs are used to help humans who speak different languages to understand each other by translating from one natural language to another. Through the years, NLP research has gone from being heavily influenced by theories of linguistics, such as those proposed by Chomsky (1959, 1965), to the corpus linguistics approach of machine learning algorithms and more recently the use of deep neural networks as neural language models such as BERT (Devlin et al. 2019) and GPT-3 (Brown et al. 2020).

According to Russell and Norvig (2010), to the contrary of formal languages, it is more fruitful to define natural language models as probability distributions over sentences rather than using definitive sets specified by grammars. The main challenges when dealing with natural languages are that they are ambiguous, large and constantly changing. That is why initial approaches to model natural languages using grammars were not as successful as modern machine learning approaches. In the former approaches, the grammars needed to be adapted and their size increased to fulfil the demands for better performance.

One important probabilistic approach to modelling natural languages involves the use of n-grams. A sequence of written symbols of length n is called an n-gram. A model of the probability distribution of strings containing n symbols is therefore called an n-gram model. This model is defined as a Markov chain of order $n-1$ in which the probability of some symbol s_i depends only on the immediately preceding $n-1$ symbols. Formally, we say $p(s_i|s_{i-1}, s_{i-2}, \ldots, s_2, s_1) = p(s_i|s_{i-1}, \ldots, s_{i-n+1})$. Probabilistic natural language models based on n-grams can be useful for text classification tasks (Russell and Norvig 2010).

Important advances in the design of algorithms for training deep neural networks, such as the recurrent long short-term memory (LSTM) network (Hochreiter and Schmidhuber 1997), have allowed researchers to move from probabilistic language models to language models based on neural networks. The LSTM neural model has been successfully applied to MT. The performance of current translator programs could not be accomplished using the approach based on language grammars alone. These new neural models are highly complex mathematical functions with thousands of parameters which are estimated iteratively from a massive number of training examples gathered from the Internet.

Some problems in NLP can be defined as MDPs and therefore they can be solved using RL algorithms. In Fig. 1, we provide a schematic example of how a RL agent would be designed to solve a language processing task in which states, actions and rewards operate mainly over strings. A set of basic operations may include appending, replacing and deleting words.

In this article, we review five main categories of such problems, namely, syntactic parsing, language understanding, text generation systems, MT, and conversational systems. Of these, conversational systems are the most studied ones, which involve finding an optimal dialog policy that should be followed by an automated system during a conversation with a human user. The other four categories are not widely known applications of reinforcement learning methods and therefore it is interesting to discuss their main benefits and drawbacks. In some of them, it is even not easy to identify the elements of a well-defined MDP. This might explain why they have not received more attention yet. Identifying these different NLP problems is important to discover new research lines at the intersection of NLP and RL.

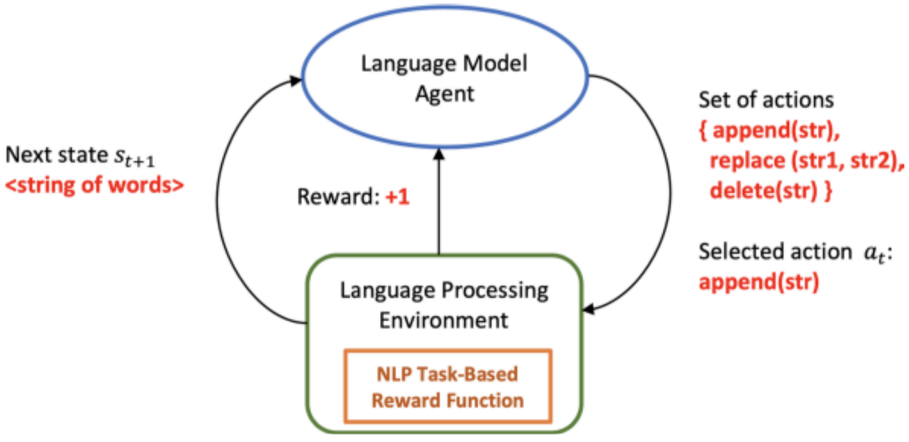

Fig. 1 Schematic view of a reinforcement learning agent designed for language processing. The language model agent acts by appending, replacing or deleting strings of words. States are strings of words. The language processing environment will provide the agent with the states and rewards after each of the interactions. The reward function is determined by the specific natural language processing task. One simple possibility for a reward function would reinforce every optimal action with a +1

In the next sections, we describe with more detail these five categories of NLP problems and their proposed solutions by means of RL. We also discuss the main achievements and core challenges on each of these categories.

2 Syntactic parsing

Syntactic parsing consists of analyzing a string made of symbols belonging to some alphabet, either in natural languages or in programming languages. Such analysis is often performed according to a set of rules called grammar. There could be many ways to perform parsing, depending on the final goal of the system (Zhang and Chan 2009; Jiang et al. 2012; Neu and Szepesvári 2009; Lê and Fokkens 2017). One of such goals could be the construction of a compiler for a new programming language when we are working with formal computer languages. Another one could be an application of language understanding for human-computer interaction.

A grammar can generate many parsing trees and each of these trees specifies the valid structure for sentences of the corresponding language. Since parsing can be represented as a sequential search problem with a parse tree as the final goal state, reinforcement learning methods are tools very well suited for the underlying sequential decision problem. In general, a parse is obtained as a path when an optimal policy is used, in a given MDP.

Consider for example the simple context-free grammar G_1 and the language $L(G_1)$ generated by it. G_1 is a 4-tuple (V, Σ, R, S) where

- $V = \{A, B\}$ is a finite set of variables,
- $\Sigma = \{0, 1, \#\}$ is a finite set, disjoint of V, containing terminal symbols,
- R is the finite set of four production rules given in Fig. 2, and
- $S \in V$ is the initial variable.

The language $L(G_1)$ generated by grammar G_1 is an infinite set of strings. Each of these strings is created by starting with the initial variable S and iteratively selecting and applying one of the production rules in G_1, also called substitution rules. For example, the string 0#1 is a valid string belonging to $L(G_1)$ and it can be generated by applying the following sequence of production rules $S \rightarrow 0A1$, $A \rightarrow B$ and $B \rightarrow \#$. Looking at this application of rules as a path of string substitutions, we have $S \Rightarrow 0A1 \Rightarrow 0B1 \Rightarrow 0\#1$. A path of substitutions, known also as derivation, can be represented pictorially as a parse tree. For example, the parse tree for the derivation of the string 00#11 is illustrated in Fig. 3.

From the previous grammar example G_1 we can notice the similarity between the elements defined in a context-free grammar $G = \{V, \Sigma, P, S\}$ and the elements defined in a MDP $M = \{S, A, T, R\}$. Let us now analyze this similarity, element by element, from the point of view of an MDP.

- The starting state s of an MDP M can be defined as the initial variable of a grammar, denoted by letter S in grammar G.

Fig. 2 Grammar G_1 with four production rules

$$\begin{aligned} \mathrm{S} &\rightarrow 0A1 \\ \mathrm{A} &\rightarrow 0A1 \mid B \\ \mathrm{B} &\rightarrow \# \end{aligned}$$

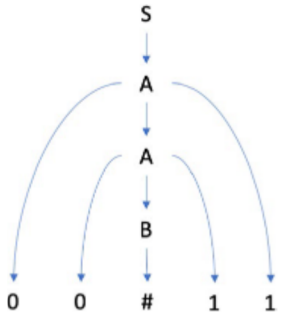

Fig. 3 Parse tree of string 00#11 generated from grammar G_1

- The set of states S in the MDP M can be defined as the set of strings generated by the grammar, in other words, the language generated by grammar G, this is $S = L(G)$.
- The set of actions A can be defined as the set of production rules given by grammar G, this is $A = R$; the MDP transition function T would be immediately defined once we have defined the set of production rules itself.
- The reward function R is the only element that cannot be taken straightforward from the elements of the grammar and it should be crafted by the designer of the system.

In the specific application of dependency parsing (Kübler et al. 2008), it has been shown that a parser can be implemented to use a policy learned by RL, in order to select the optimal transition in each parsing stage (Zhang and Chan 2009). Given a sentence with n words $x = w_1 w_2 \dots w_n$, we can construct its dependency tree by selecting a sequence of transitions. A stack data structure is used to store partially processed words and also a queue data structure is used to record the remaining input words together with the partially labeled dependency structure constructed by the previous transitions. The construction of the dependency tree is started with an empty stack and the input words being fed into the queue. The algorithm performs four different types of transitions until the queue is empty. These four transitions are: *reduce*, which takes one word from the stack; *shift*, which pushes the next input word into the stack; *left-arc*, which adds a labeled dependency arc from the next input word to the top of the stack and then takes the word from the top of the stack; and finally *right-arc*, which adds a dependency arc from the top of the stack to the next input word and pushes that same word into the stack. During the construction of the parsing tree each one of the transitions is selected using a reward signal. In this particular implementation the optimal policy for selecting the transitions is estimated using the SARSA RL algorithm.

An interesting modification found in the implementation of this algorithm is the replacement of the Q function by an approximation computed through the calculation of the negative free energies of a restricted Boltzmann machine. The results of this approach for dependency parsing using RL are comparable with state-of-the-art methods. More recently, it has been shown that RL can also be used to reduce error propagation in greedy dependency parsing (Lê and Fokkens 2017). In another approach, Neu and Szepesvári (2009) used a number of inverse RL (IRL) algorithms to solve the parsing problem with probabilistic context-free grammars. In IRL, given a set of trajectories in the environment, the goal is to find a reward function such that if it is used for estimating the optimal policy, the resulting policy can generate trajectories very similar to the original ones (Ng and Russell 2000).

Another dual learning approach for solving the semantic parsing problem is presented by Cao et al. (2019). This dual learning algorithm follows the same strategy used by Zhu et al. (2020), consisting of an adversarial training scheme that can use both labeled and

unlabeled data. The primary task (semantic parsing) learns the transformation from a query to logical form (Q2LF). The secondary task (natural language generation) learns the transformation from a logical form to a query (LF2Q). The agent from the primary task can teach the agent from the secondary task and vice versa in a RL fashion. A validity reward by checking the output of the primary model at the surface and at semantic levels is used. This reward function requires prior knowledge of the logical forms of the domain of interest, and it is used to check for completeness and well-formed semantic representations. The experimental results showed that semantic parsing based on dual learning improves performance across datasets.

In a probabilistic context-free grammar, each production rule has a probability assigned to it, which results in the generation of expert trajectories. Speeding up the learning of parse trees using RL has also been studied, specifically the use of apprenticeship RL as a variation of IRL has been shown to be an effective method for learning a fast and accurate parser, requiring only a simple set of features (Jiang et al. 2012). By abstracting the core problem in syntactic parsing, we can clearly see that it can be posed as an optimization problem in which the input is a language grammar G and one input string w_1 to be parsed, and the output is a parse tree that allows the correct parsing of w_1. This problem gives rise to the following MDP (S, A, T, R) (Lê and Fokkens 2017):

- The set of states S is defined as the set of all possible partial or complete parse trees that can be generated with the given grammar G and the string w_1.
- The set of actions A is formed with all the grammar rules contained in G, this is, the application of each derivation rule of the grammar is considered to be an action.
- The transition function T can be completely determined and it is deterministic, because given a selected grammar rule and the current partially parsed string, we can know with certainty the next resulting intermediate parse tree of that string.
- Finally, the reward function R can be defined as a function of the number of arcs that are correctly labeled in the resulting parse tree.

Based on this MDP we can formulate a RL system as illustrated in Fig. 4.

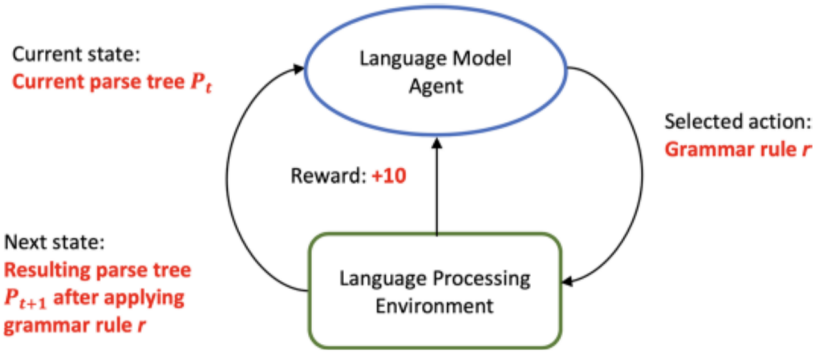

Fig. 4 Schematic view of a reinforcement learning agent designed for syntactic parsing. The language processing environment will provide the agent with the states and rewards after each of the interactions. The reward function can be defined in various ways, for example, a positive reward of 10 may be provided each time an appropriate grammar rule is applied

Fig. 5 Grammar defining valid sentences in English, grammar adapted from Sipser (2013)

⟨SENTENCE⟩ → ⟨NOUN_PHRASE⟩⟨VERB_PHRASE⟩
⟨NOUN_PHRASE⟩ → ⟨CMPLX_NOUN⟩ | ⟨CMPLX_NOUN⟩⟨PREP_PHRASE⟩
⟨VERB_PHRASE⟩ → ⟨CMPLX_VERB⟩ | ⟨CMPLX_VERB⟩⟨PREP_PHRASE⟩
⟨PREP_PHRASE⟩ → ⟨PREP⟩⟨CMPLX_NOUN⟩
⟨CMPLX_NOUN⟩ → ⟨ARTICLE⟩⟨NOUN⟩
⟨CMPLX_VERB⟩ → ⟨VERB⟩ | ⟨VERB⟩⟨NOUN_PHRASE⟩
⟨ARTICLE⟩ → a | the
⟨NOUN⟩ → customer | discount | refund
⟨VERB⟩ → wants | requests | cancelled
⟨PREP⟩ → with

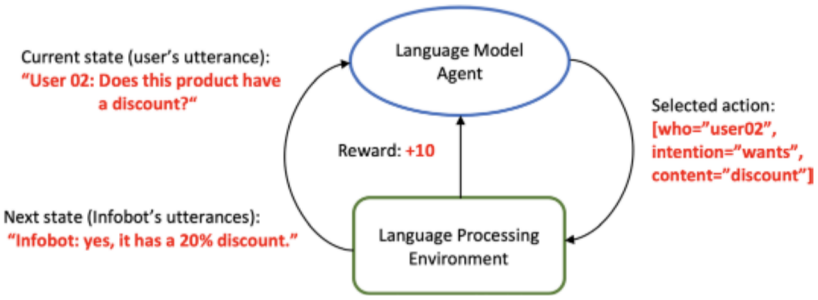

Fig. 6 Schematic view of a reinforcement learning agent designed for text understanding, as an example application. The language model agent acts by assigning values to a vector or list of variables, as a function of the utterance of a user. The current user *user*02 is computed from the user's utterance. The next state is the string generated from the vector of variables as understood by the agent, for example, using a text generator agent (see Fig. 7). The language processing environment will provide the agent with the states and rewards after each of the interactions. The environment and the reward function are determined by the language understanding task being solved, i.e., an infobot

3 Language understanding

Language understanding can also be posed as a MDP and therefore we can apply sophisticated reinforcement learning algorithms designed in recent years. Furthermore, we can implement them together with deep neural networks to cope with the massive amount of data that text understanding applications typically require.

Consider a problem of natural language understanding (NLU). In such a problem we could have a grammar like the one given in Fig. 5 that allows a program to automatically determine the elements of a sentence written in English. Using this grammar, sentences such as "The customer with a discount wants a refund" and "The customer with a discount cancelled the refund" can be analyzed by an automated system to determine the intention of the customer, which in this case is whether she wants a refund or she wants to cancel a refund she had previously requested. Therefore, a grammar can be used to detect users' intentions while reinforcement learning can be used to select the optimal sequence of substitutions during the parsing process of the input sentences. Once the parser program has been used to determine the grammatical role of each word in the input text string, the result can be stored in a vector-type structure such as [*who* = user02, *intention* = "wants", *content* = "discount"]. This vector-type representation of variables *who*, *intention* and *content*, can be used for another program to determine the most appropriate action to be performed next. For example, informing about a discount to a customer. Figure 6 outlines the procedure.

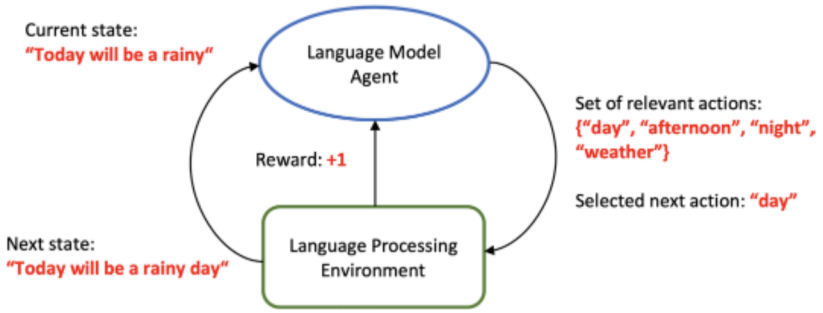

Fig. 7 Schematic view of a reinforcement learning agent designed for language generation, as an example application. The language model agent acts by selecting words from a relevant set of words, which is a function of the current state. The current state is a—possibly incomplete—sentence in English. The next state is the sentence resulting from appending the word selected by the agent. The language processing environment will provide the agent with the states and rewards after each of the interactions. Actions might take the form of strings of characters such as n-grams, words, sentences, paragraphs or even full documents. The environment and the reward function are determined by the language processing task being solved, i.e., text generation

Ambiguities are an important problem in language understanding. For example, the sentence "the child observes the cat in the tree" may have two interpretations, whether the child is in the tree or the cat is in the tree. This kind of ambiguity in the language is hard to solve even by humans. Sometimes it can be solved by using context or common sense. From the point of view of RL, there is no obvious way to solve it either. One approach to this problem would be to leverage the powerful text embedding vectors generated by sophisticated language models such as GPT together with a function that rewards making corrections as learning interactions go on, taking advantage of the context. GPT-based models are very good at keeping contextual information. A reward function could provide a larger reward when the interpretation of the intent is more highly evaluated by a context metric provided by the language model.

Language understanding programs approached by RL have to deal with systems that automatically interpret text or voice in the context of a complex control application, and use the knowledge extracted to improve control performance. Usually, the text analysis and the learning of the control strategy are carried out both at the same time. For example, Vogel and Jurafsky (2010) implement a system capable to learn to execute navigational instructions expressed in a natural language. The learning process is carried out using an apprenticeship approach, through pairs of paths in a map and their corresponding descriptions in English. The challenge here is to discover which commands match English instructions for navigation. The correspondence is learned applying RL and using the deviation between the given desired path and the route being followed for the reward signal. This work demonstrates that the semantic meaning of spatial terms can be grounded into geometric properties of the paths. In a similar approach to language grounding (Branavan et al. 2012) the system learns to interpret text in the context of a complex control application. Using this approach, text analysis and control strategies are learned jointly using a neural network and a Monte Carlo search algorithm. The approach is tested on a video game, using its official manual as a text guide.

DRL has also been used to automatically play text games (He et al. 2016), showing that it is possible to extract meaning rather than simply memorizing strings of texts. This is also the case of the work presented by Guo et al. (2017), where an LSTM and a DQN are employed to solve the sequence-to-sequence problem. This approach is tested with

the problem of rephrasing a natural language sentence. The encoding is performed using the LSTM and the decoding is learned by the DQN. The LSTM initially suggests a list of words which are taken by the DQN to learn an improved rephrasing of the input sentences.

Zhu et al. (2020) presented a semi-supervised approach to tackle the dual task of intent detection and slot filling in NLU. The suggested architecture consists of a dual pseudo-labeling method and a dual learning algorithm. They apply the dual learning method by jointly training the NLU and semantic-to-sentence generation (SSG) models, using one agent for each model. As the feedback rewards are non-differentiable, a RL algorithm based on policy gradient is applied for optimization. The two agents collaborate in two closed loops. The NLU2SSG loop starts from a sentence, first generating a possible semantic form by the NLU agent and then reconstructing the original sentence by SSG. The SSG2NLU loop goes in reverse order. Both the NLU and SSG models are pre-trained on labeled data. The corresponding validity rewards for the NLU and SSG evaluate whether the semantic forms are valid. The approach was evaluated on two public datasets, i.e., ATIS and SNIPS, achieving state-of-the-art performance. The proposed framework is agnostic of the backbone model of the NLU task.

Text understanding is one of the most recent natural language problems approached using RL, specifically by DRL. This approach consists of mapping text descriptions into vector representations. The main goal is to capture the semantics of the texts. Therefore, learning good representations is key. In this context, it has been argued that LSTMs are better than bag-of-words (BOW) when combined with RL algorithms. The reason is that LSTMs are more robust to small variations of word usage, and they can learn some underlying semantics of the sentences (Narasimhan et al. 2015).

As we have seen above, the main applications of reinforcement learning in the context of language understanding have been focused on the learning of navigational directions. RL or IRL recommend themselves over supervised learning due to the good match between sequential decision making and parsing. However, it is not difficult to think of other similar applications that could take advantage of this approach. For example, if we can manage to design a system capable to understand text to some degree of accuracy, such a system could be used to implement intelligent tutors, smart enough to understand the questions posed by the user and select the most appropriate learning resource, whether it is some text, audio, video, hyperlink, etc.

Interestingly, the successful results recently obtained with the combination of deep neural networks and RL algorithms open another dimension of research that appears to be promising in the context of parsing and text understanding. As we have mentioned before, creating natural language models is difficult because natural languages are large and constantly changing. We think that DRL could become the next best approach to natural language parsing and understanding. Our reasoning is based primarily on two facts. First, DRL can store optimally thousands of parameters of the grammars as a neural model, and we have already evidence that these neural models can be very effective with other natural language problems such as machine translation. Second, RL methods would allow the agent to keep adapting to changes in a natural language, since the very nature of these algorithms is to learn through interaction and this feature allows the RL agents to constantly adapt to changes in their environment.

4 Text generation systems

Text generation systems are built to automatically generate valid sentences in natural language. One of the components of such systems is a language model. Once the language model is provided or learned, the optimization problem consists of generating valid sequences of substrings that will subsequently complete a whole sentence with some meaning in the domain of the application.

Given a vector representation of a set of variables in a computational system and their corresponding values, a reinforcement learning algorithm can be used to generate a sentence in English, or any other natural language, that can serve to communicate specific and meaningful information to a human user. However, using the information stored in a set of program variables and constructing sentences in a natural language representing such information is not an easy task. This problem has been studied in the context of generating navigational instructions for humans, where the first step is to decide about the content that the system wants to communicate to the human, and the second step is to build the correct instructions adding word by word. An interesting point in this approach is that the reward function is implemented as a hidden Markov model (Dethlefs and Cuayáhuitl 2011) or as a Bayesian network (Dethlefs and Cuayáhuitl 2011). The RL process is carried out with a hierarchical algorithm using semi-MDP's.

Text generation has also been approached using IRL (Ziebart et al. 2008) and generative adversarial networks (GANs, Goodfellow et al. 2014). Shi et al. (2018) proposed a new method combining GANs and IRL to generate text. The main result of this work is the alleviation of two problems related to generative adversarial models, namely reward sparsity and mode collapse. The authors of this work also introduced new evaluation measures based on BiLingual Evaluation Understudy (BLEU) score, designed to evaluate the quality of the generated texts in terms of matching human-generated expert translations. They showed that the use of IRL can produce more dense reward signals and it can also generate more diversified texts. With this approach, the reward and the policy functions are learned alternately, following an adversarial model strategy. According to the authors, this model can generate texts with higher quality than previous proposed methods based also on GANs, such as SeqGAN (Yu et al. 2017), RankGAN (Lin et al. 2017), MaliGAN (Che et al. 2017) and LeakGAN (Guo et al. 2018). The adversarial text generation model uses a discriminator and a generator. The discriminator judges whether a text is real or not, meanwhile the generator learns to generate texts by maximizing a reward feedback provided by the discriminator through the use of RL. The generation of entire text sequences that these adversarial models can accomplish helps to avoid the exposure bias problem, a known problem experienced by text generation methods based on RNNs. The exposure bias problem (Bengio et al. 2015) lets small discrepancies between the training and inference phases accumulate quickly along the generated sequence.

In a text generation task the corresponding MDP might be defined as follows:

- Each state in S is formed with a feature vector describing the current state of the system being controlled, containing enough information to generate the output string. We can visualize this feature vector as a set of variables that describe the current status of the system.
- Actions in A will consist of adding or deleting words.
- With respect to the transition function T, every next state can be determined by the resulting string, after we have added or deleted a word.

– In this task, the reward function could be learned from a corpus of labeled data or more manually, from human feedback.

An advantage of RL methods over supervised learning for text generation becomes apparent when there is a diversity of valid text output, i.e., multiple different generations would be of equal quality. In this case, it is problematic for supervised learning to define a differentiable error for backpropagation. However, evaluation measures like BLEU or the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) can be used well to define a reward function for RL (Keneshloo et al. 2020). Future research work can focus on adaptive natural language generation during human–computer interaction, assuming a continuously changing learning environment. In natural language generation the main goal is to build a precise model of the language, and the current existing approaches are far from being generic.

Another more complicated possibility is the study of language evolution under a RL perspective. In general, language evolution is concerned with how a group of agents can create their own communication system (Cangelosi and Parisi 2002). The communication system emerges from the interaction of a set of agents inhabiting a common environment. A process like this can be modeled as a RL multi-agent system (Mordatch and Abbeel 2018).

Li et al. (2018) used RL and IRL for paraphrase generation. One of the components of this approach is a generator. The generator is initially trained using deep learning and then it is fine-tuned using RL. The reward of the generator is given by a second component of the architecture, the evaluator. The evaluator is a deep model trained using IRL to evaluate whether two given phrases are similar to each other.

5 Machine translation

MT consists in automatically translating sentences from one natural language to another one, using a computing device (Hutchins and Somers 1992). An MT system is a program that receives text (or speech) in some language as input and automatically generates text (or speech), with the same meaning, but in a different language (see Fig. 8). Early MT systems translate scientific and technical documents, while current developments involve

Fig. 8 Schematic view of a reinforcement learning agent designed for language translation. It gets as input a text in some language A, and responds with another text string in a different language B. Input and output text strings have the same meaning. The language model agent acts by selecting the most relevant string of words. The language processing environment will provide the agent with the states and rewards after each of the interactions. The environment and the reward function are determined by the machine translation task being solved, i.e., translation from English to Spanish

online translation systems, teaching systems, among others. MT systems have been successfully applied to an increasing number of practical problems (Way 2018). Since 1949, when the task of MT was proposed to be solved using computers (Weaver 1955), several approaches have been studied over the years.

Statistical MT (SMT) is by far the most studied approach to MT. In this paradigm, translations are generated using statistical models whose parameters are estimated through the analysis of many samples of existing human translations, known as bilingual text corpora (Brown et al. 1990; Koehn 2009; Williams et al. 2016). SMT algorithms are characterized by their use of machine learning methods, where neural networks have been used with some success (Cho et al. 2014; Devlin et al. 2014; Kalchbrenner and Blunsom 2013).

In the last decade neural networks have won the battle against statistical methods in the field of translation. Neural MT (NMT, Stahlberg 2020) uses large neural networks to predict the likelihood of a sequence of words. NMT methods have been broadly applied to advance up-to-date phrase-based SMT systems, where a unit of translation may be a sequence of words (instead of a single word), called a phrase (Koehn et al. 2003). NMT systems became a major area of development since the emergence of deep neural networks in 2012 (Bahdanau et al. 2015; Wu et al. 2016; He et al. 2017; Hassan et al. 2018; Lam et al. 2019). Current state-of-the-art machine learning translation systems rely heavily on recurrent neural networks (RNN), such as the LSTM network (Hochreiter and Schmidhuber 1997). In the sequence-to-sequence approach (Sutskever et al. 2014) depicted in Fig. 9, which was used for translation (Wu et al. 2016), two RNNs are needed, an encoder and a decoder. The encoder RNN updates its weights as it receives a sequence of input words in order to extract the meaning of the sentence. Then, the decoder RNN updates its corresponding weights to generate the correct sequence of output words, in this case, the translated sentence. In the RNN approach the encoder makes reference to a program that would internally encode or represent the meaning of the source text, meanwhile the decoder will decode that internal representation and output a translated sentence with the correct meaning. There are two problems that arise in the training and testing of seq2seq models. These problems are known as (1) exposure bias, i.e., the discrepancy between ground-truth dependent prediction during training and model-output dependent prediction during testing, and (2) inconsistency between the training and test objectives, i.e., measurement. Both problems have been recently studied and various solutions based on RL have been proposed (Keneshloo et al. 2020).

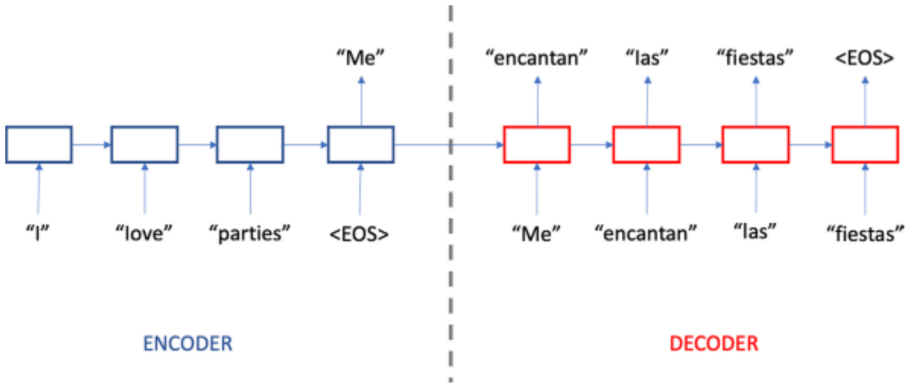

Fig. 9 Sequence-to-sequence RNN architecture for machine translation, adapted from Sutskever et al. (2014)

Similarly to what can be accomplished in conversational systems, in MT, we see that RL algorithms can be used to predict the next word or phrase to be uttered by a person, specially during a simultaneous translation task, where the content is translated in real-time as it is produced (Fügen et al. 2007). This prediction is useful to increase the quality and speed up the translation.

In the case of the training, when it is done interactively, there is evidence that RL can be used to improve the real-time translation performance after several interactions with humans (Grissom II et al. 2014; Sokolov et al. 2015, 2016). Gu et al. (2017) propose an NMT model for real-time translation, where a task-specific neural network learns to decide which actions to take (i.e., to wait for another source word or to emit a target word) using a fixed pre-trained network and policy gradient techniques. Furthermore, to tackle the need of massive training data in MT, He et al. propose a dual learning mechanism, which automatically learns from unlabeled data (He et al. 2016). This method is based on the fact that using a policy gradient algorithm together with a reward function defined as the likelihood of the language model, it is possible to create a translation model using examples of translation going in both directions, from language one to language two, and from language two to language one. With this approach it is possible to obtain an accuracy similar to the accuracy obtained with other neural models, but using only 10% of the total number of training examples.

Speech translation systems have improved recently due to simultaneous MT, in which translation starts before the full sentence has been observed. In traditional speech translation systems, speech recognition results are first segmented into full sentences, then MT is performed sentence-by-sentence. However, as sentences can be long, i.e., in the case of lectures or presentations, this method can cause a significant delay between the speaker's utterance and the translation results, forcing listeners to wait a noticeable time until receiving the translation. Simultaneous MT avoids this problem by starting to translate before the sentence boundaries are detected. As a first step in this direction, Grissom II et al. (2014) propose an approach that predicts next words and final verbs given a partial source language sentence by modeling simultaneous MT as a MDP and using RL. The policy introduced in this method works by keeping a partial translation, querying an underlying MT system and deciding to commit these intermediate translations occasionally. The policy is learned through the iterative imitation learning algorithm SEARN (Daumé et al. 2009). By letting the policy predict in advance the final verb of a source sentence, this method has the potential to notably decrease the delay in translation from languages in which, according to their grammar rules, the verb is usually placed in the end of the phrases, such as German. However, the successful use of RL is still very challenging, especially in real-world systems using deep neural networks and huge datasets (Wu et al. 2018).

RL techniques have also had a positive impact in SMT, which uses predictive algorithms to teach a computer how to translate text based on creating the most probable output learned from different bilingual text corpora. As the goal in RL is to maximize the expected reward for choosing an action at a given state in an MDP model, algorithms based on bandit feedback for SMT can be visualized as MDP's with one state, where selecting an action represents the prediction of an output (Langford and Zhang 2007; Li et al. 2010). Bandit feedback inherits the name from the problem of maximizing the amount of rewards obtained after a sequence of plays with a one-armed bandit machine, without a priori knowledge of the reward distribution function of the bandit machine. Sokolov et al. (2015) propose a structured prediction in SMT based on bandit feedback, called *bandit expected loss minimization*. This approach uses stochastic optimization for learning from partial feedback in the form of an expected 1-BLEU loss criterion (Och 2003; Wuebker

et al. 2015), as opposed to learning from a gold standard reference translation. This is a non-convex optimization problem, which they analyzed in the stochastic gradient method of pseudogradient adaptation (Poljak 1973) that allowed to show convergence of the algorithm. Nevertheless, the algorithm of Sokolov et al. (2015) presents slow convergence. In other words, such a system needs many rounds of user feedback in order to learn in a real-world SMT. Moreover, it requires absolute feedback of translation quality. Therefore, Sokolov et al. (2016) propose improvements with a strong convexification of the learning objective, formalized as bandit cross-entropy minimization to overcome the convergence speed problem. They also propose a learning algorithm based on pairwise preference rankings, which simplifies the feedback information.

The same approach used for MT can be used in a rephrasing system (Guo 2015). This system receives a sentence as an input, creates an internal representation of the information contained in such a sentence and then generates a second sentence with the same meaning of the first one. The algorithms used to solve such a challenging problem are the LSTM and a DQN. The former is used to learn the representation of the input sentence and the latter is used to generate the output sentence. The experiments presented in this work indicate that the proposed method performs very well at decoding sentences. Furthermore, the algorithm significantly outperformed the baseline when it was used to decode sentences never seen before, in terms of BLEU scores. The generation of the output string is not explicitly computed from a vector of variables, instead, this vector representation is implicitly learned and stored in the weights of the LSTM and the DQN. Similarly, this system does not need an explicit model of the language to do the rephrasing, because that model is also learned and stored in its neural networks. Therefore, the inner workings of this system are the same as a machine learning translator. It receives a string of words as input and generates another string of words with the same meaning.

The rephrasing problem aforementioned consists in generating one string B based on some input string A, in such a way that both strings have the same meaning. Considering this task we can define an MDP (S, A, P, R) as proposed in Guo (2015):

- The set of states S is defined as the set of all possible input strings w_i.
- The set of actions A consists of adding and deleting words taken from some vocabulary.
- The transition function P can be completely determined and it is deterministic. The next state is the string that results from adding or deleting a word.
- Finally, the reward function R can be defined as a function that measures how similar the strings A and B are, in semantical terms.

In general, MT can be defined as an optimization problem. In the particular case of simultaneous translation, we can define an MDP (S, A, P, R) and solve it using RL as we explain next. Given an utterance in a language A, we need to find the optimal utterance B that maximizes a measure of semantic similarity with respect to A. In this kind of translation problem, when the sentences need to be translated as fast as possible, RL can be used for learning when a part of a sentence should be trusted and used to translate future parts of the same sentence. In this way the person waiting for the translated sentence does not need to wait until the translator gets the last word of the original sentence to start the translation process. Therefore, the translation process can be accelerated by predicting the next noun or verb. The corresponding MDP is the following (Grissom II et al. 2014):

- Each state in S contains the string of words already seen by the translator and the next predicted word.

- The actions in A are mainly of three types: to commit to a partial translation, to predict the next word, or to wait for more words.
- The transition function P, indicating the transitions from one state to another is fully determined by the current state and the action performed. We can compute the resulting string after applying an action.
- The reward function R can be defined based on the BLEU score (Papineni et al. 2002), which basically measures how similar one translation is compared to a reference string, which is assumed to be available for training.

There is a number of improvements that could be researched in simultaneous MT using RL. One is the implementation of these systems in more realistic scenarios where faster convergence is required. Currently, the experimentation with this approach has involved idealized situations in which the phrase to be translated contains only one verb. This constraint should be dropped if we want to employ them in real-world scenarios.

Experiments with other languages are also needed, especially for those languages that do not fall into the set of most spoken languages in the world. This will require the estimation of different optimal MDP policies, one for each language. However, if the correct recurrent neural model can be defined, using RL might help in autonomously learning machine translation. In the same way that AlphaGo managed to play multiple games against itself and improved in the process, it might be the case that future translator algorithms can learn multiple natural languages by talking to themselves.

6 Conversational systems

Conversational systems are designed to interact with various users using natural language, most commonly in verbal or written form. They are well structured and engineered to serve for instance as automated web assistance or for natural human–robot interaction. The architecture and functionality of such systems are heavily dependent on the application.

There are two classes of conversational systems. First, open domain systems, usually known as chatbots. They are built in a Turing-test fashion. This is, they can hold a conversation basically about any topic, or at least they are trained with that goal in mind. Second, closed domain systems which are developed more as expert systems, in the sense that they should serve a conversational purpose very well defined and bounded. They should be able to provide information or assistance about a specific topic. In this article we are more interested in this latter system, since serving a well-defined task, can more easily benefit from reinforcement learning, due to reduced state and action spaces.

In this section, we will see that RL algorithms can be used to generate suitable responses during a conversation with a human user. If the system can be programmed to predict with some accuracy how a conversation might occur, then it can optimize the whole process in such a way that the system can provide more information in less interactions if we are talking about a system designed to inform humans, or it can make a more interesting conversation if it is designed as a chatbot for entertainment. There are a number of factors that affect the effectiveness of a conversational system, including context identification, dynamic context adaptation, user intention (Crook et al. 2014), and domain knowledge (Higashinaka et al. 2015).

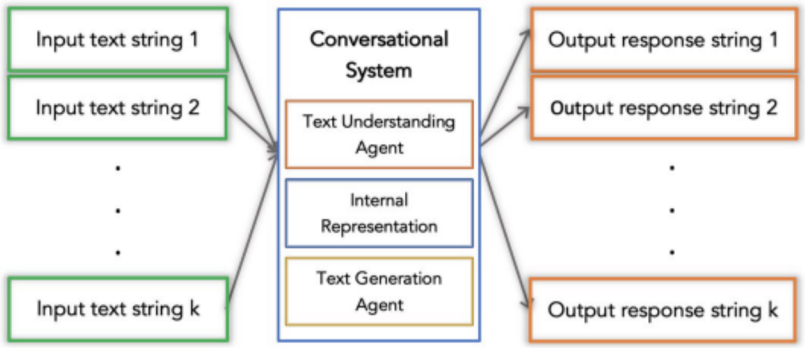

Fig. 10 Information flow of a conversational system. This system receives as input a text string containing a question or simply a comment, and it responds with another text string containing the response. This input and response interaction typically iterates several times. Going from "Input text string x" to "Output response string x" requires the application in sequence of a text understanding agent (see Fig. 4) and a text generator agent (see Fig. 7)

Conversational systems consist of three basic components whose sophistication will vary from system to system. These components are (Fig. 10):

1. processing of the input message (perception),
2. the internal state representation (semantic decoder), and
3. the actions (dialogue manager).

The input is a message from the user, for instance, speech, gestures, text, etc. The user's input message is converted to its semantic representation by the semantic encoder. The semantic representation of the message is further processed to determine an internal state of the system from which the next action is determined by the dialogue manager. Finally, the actions might include the generation of natural speech, text or other system actions.

Conversational systems are often heuristically-driven and thus the flow of conversation as well as the capabilities are specifically tailored to a single application. Application-specific rule-based systems can achieve reasonably good performance due to the incorporation of expert domain knowledge. However, this often requires a huge number of rules, which becomes quickly intractable (Higashinaka et al. 2015).

Due to the limitations of rule-based systems there are ongoing efforts to use data-driven or statistical conversational systems based on RL since the early 2000s (Litman et al. 2000; Levin et al. 2000; Singh et al. 2000; Singh et al. 2002; Walker 2000; Young 2000). In theory, these data-driven conversational systems are capable of adapting based on interactions with real users. Additionally, they require less development effort but at a cost of significant learning time. Although very promising they still need to overcome several limitations before they are adopted for real-world applications. These limitations stem from both the problem itself and from RL algorithms.

RL could potentially be applied to all three components of a conversational system mentioned above, starting with perception of the input message, internal system representations as well as the decision of the system's output. However, we argue that RL is more readily available for improving the dialogue manager which deals directly with the user interaction. More difficult but also possible using DRL would be the learning of suitable internal representations based on the success of the interactions.

In a recent survey on neural approaches to conversational AI (Gao et al. 2018), it is recognized that in the last few years, RL together with deep learning models have helped to significantly improve the quality of conversational agents in multiple tasks and domains. Key aspects of this combination of learning models are that conversational systems are allowed to adapt to different environments, tasks, domains and even user behaviors.

A large body of research exists for RL-based conversational systems. For instance, POMDP-based conversational systems (Williams and Young 2007; Young et al. 2010; Thomson and Young 2010; Young et al. 2013; Crook et al. 2014) emerged as a strategy to cope with uncertainty originating from the perceptual and semantic decoder components. However, they also suffer from very large state representations that often become intractable (*curse of dimensionality*) which typically necessitates some sort of state space compression (Crook et al. 2014). We attribute this limitation to the widespread use of discrete state space representations typical in dialogue management and early days of RL algorithms. We believe that such limitation could be overcome with continuous state space representations and the use of function approximation techniques such as DQN (Mnih et al. 2015), VIN (Tamar et al. 2016), A3C (Mnih et al. 2016), TRPO (Schulman et al. 2015) and many others. Although there have been attempts to use function approximation techniques within dialogue management systems (Jurcicek et al. 2010; Henderson et al. 2008), these have not been scaled up. Li et al. (2016) simulated a dialogues between two virtual agents, and sequences that display three useful conversational properties are rewarded. These properties are: informativity, coherence, and ease of answering. This RL model uses policy gradient methods.

The main implications of using a continuous representation of the states is that we are required to estimate less parameters than when we use a discrete state representation. This is the case when we are dealing with large state spaces. As a result of handling less parameters the learning of policies can be significantly accelerated. Moreover, the quality of the learned policies is usually better than the policies learned with discretized state spaces. When we are implementing DRL models the number of weights in our neural network used to store the value functions can be large. However, the number of parameters of a deep model is less than the number of discrete states for which we would need to estimate a value.

Lemon (2011) showed that natural language generation problems can be solved using RL by jointly optimizing the generation of natural language and the management of dialogues. Another approach based on RL to improve the long-turn coherence and consistency of a conversation is proposed in Yu et al. (2017). With this approach it is possible to obtain smooth transitions between task and non-task interactions. Papaioannou and Lemon (2017) present a chatbot system for task-specific applications. This system for multimodal human–robot interaction can generate longer conversations than a rule-based algorithm. This implies that the learned policy is highly successful in creating an engaging experience for chat and task interactions. A conversational agent can be effectively trained using a simulator (Li et al. 2017). After a preliminary training, the agent is deployed in the real scenario in order to generate interactions with humans. During these interactions with the real world the agent keeps learning. In a similar approach, Li et al. (2017) used a movie booking system to test a neural conversational system trained to interact with users by providing information obtained from a structured database. Interestingly, if the action spaces of the agents are treated as latent variables, it is possible to induce those action spaces from the available data in an unsupervised learning manner. This approach can be used to train dialogue agents using RL (Zhao et al. 2019).

Some researchers have tried to develop question answering (QA) systems with multi-step reasoning capabilities, based on RL. Though QA systems cannot be considered full conversational systems, both share some common challenges. DeepPath (Xiong et al. 2017), MINERVA (Das et al. 2018) and M-Walk (Shum et al. 2018) are recent examples of systems that perform multi-step reasoning on a knowledge base through the use of RL.

More recently, Yang et al. (2021) presented a dialogue system that learns a policy that maximizes a joint reward function. The first reward term encourages topic coherence by computing the similarity between the topic representation of the generated response and that of the conversation history. The second term encourages semantic coherence between the generated response and previous utterance by computing mutual information. The last term is based on a language model to estimate the grammatical correctness and fluency of the generated response. Lu et al. (2019) used Hindsight Experience Replay (HER) to address the problem of sparse rewards in dialogues. HER allows for learning from failures and is thus effective for learning when successful dialogues are rare, particularly early in learning. Liu et al. (2020) showed that the goal is to model understanding between interlocutors rather than to simply focus on mimicking human-like responses. To achieve this goal, a transmitter–receiver-based framework is proposed. The transmitter generates utterances and the receiver measures the similarity between the built impression and the perceived persona. Mutual persona perception is then used as a reward to learn to generate personalized dialogues. Chen et al. (2020) proposed a structured actor-critic model to implement structured DRL. It can learn in parallel from data taken from different conversational tasks, achieving stable and sample-efficient learning. The method is tested on 18 tasks of PyDial (Ultes et al. 2017). Papangelis et al. (2019, 2020) presented a complete attempt at concurrently training conversational agents. Such agents communicate only via self-generated language, outperforming supervised and deep learning baselines. Each agent has a role and a set of objectives, and they interact using only the language they have generated.

One major problem regarding the building of conversational systems lies in the amount of training data needed (Cuayáhuitl et al. 2014) which could originate from simulations (as in most of the research), offline learning (limited number of interaction data sets) and learning from interactions with real users. In fact, training and evaluating such systems require large amounts of data. Similarly, measuring the performance of conversational systems is itself a challenge and different ways of measuring it have been proposed. One way is based on the use of some predefined metrics that can be used as the reward function of the system, for example, some measurement of the success rate of the system, which can be calculated when the system solves the user's problem. Another way of giving reward to the system is by counting the number of turns, which gives preference to more succinct dialogues. A more sophisticated way would be to automatically assess the sentiment of the evolving conversation, generating larger rewards for positive sentiment (Bothe et al. 2017). Other metrics that are being explored are the coherence, diversity and personal style of a more human-like conversational system (Gao et al. 2018).

Another way of measuring the performance is through the use of human simulators. However, programming human simulators is not a trivial task. Moreover, once we have found a functional dialogue policy, there is no way to evaluate it without relying on heuristic methods. Some simulators are completely built from available data. The way they work is basically by selecting at the start of each training episode a randomly generated goal and a set of constraints. The performance of the system is measured by comparing the sequence of contexts and utterances generated after each step during the training. User simulation is not obvious and is still an ongoing research field.

In general, conversational systems can be classified into two different types: (1) task-oriented systems, and (2) non-task-oriented systems. Both types of systems can be defined as a general optimization problem that can be solved using RL algorithms. An MDP (S, A, T, R) with the main elements required to solve such an optimization problem is the following:

- The set of states S is defined as the history of all utterances, such as comments, questions and answers happening during the dialogue.
- The set of actions A consists of all the possible sentences that the system can answer to the user in the next time step.
- The transition function T. The next state is the updated history of utterances after adding the last sentence generated by the system or the user. The transition function is non-deterministic in the case of non-predictable user responses.
- Finally, the reward function R can be defined as a function that measures the performance of the system, or how similar the generated dialogue is with respect to a reference dialogue from an existing corpus.

The training of conversational systems could be also done using human users or using a model learned from corpora of a human–computer dialogue. However, the large number of possible dialogue states and strategies makes it difficult to be explored without employing a simulator. Therefore, the development of reliable user simulators is imperative for building conversational systems, and this comes with its own set of challenges.

Simulators are in particular useful for getting effective feedback from the environment during learning. For instance, Schatzmann and Young (2009) implemented a user simulator using a stack structure to represent the states. The dialogue history in this approach consists of sequences of push and pop operations. Experiments show the effectiveness of this method to optimize a policy and it was shown to outperform a hand-crafted baseline strategy, in a real-world dialogue system. However, using a simulator always has serious limitations, whether it is manually coded, learned from available data, or a mixture of these approaches. A simulator is by definition not the real environment and therefore a RL policy trained on it will need some or many adjustments to make it work properly in the real environment. In general, the development of realistic simulators for RL and the related methodologies to fine-tune the policies afterwards to make them generalize well in the real world is still an open question. Moreover, the reward function is key to providing effective feedback. It is well known that the design of reward functions is a challenging task that requires expert knowledge on the task to be learned and on the specific algorithm being used. Very often, it is only after many iterations in the design process and a significant amount of experimentation that reward functions are optimally configured. Su et al. studied reward estimation (Su et al. 2018). This approach is based on the one hand on the use of a RNN pre-trained off-line to serve as a predictor of success and on the other hand, a dialogue policy and a reward function are trained together. The reward function is modeled with a Gaussian process using active learning.

Chen et al. propose an interactive reinforcement learning framework to address the cold start problem (Chen et al. 2017). The framework, referred to as a companion teacher, consists of three parties: (1) one learning agent, (2) a human user, and (3) a human 'companion' teacher. The agent (dialogue manager) consists of a dialogue state tracker and a policy model. The human teacher can guide learning at every turn (time step). The teacher can guide learning by both reward or policy-shaping. The authors assume that the dialogue states and policy model are visible to the human teacher. In follow-up work (Chen et al. 2017), a rule-based system is used for reward- and policy-shaping, but the same strategy could be used to incorporate

human feedback. The learning agent is implemented using a DQN and two separate experience memories for the agent and teacher. Uncertainty estimation is used to control when to ask for feedback and learn from the experience memories. Simulation experiments showed that the proposed approach could significantly improve learning speed and accuracy.

7 Other language processing tasks

RL has also been used for the improvement of information extraction through the acquisition and incorporation of external information (Narasimhan et al. 2016). In this work, a DQN is trained to select actions based on contextual information, leading the information retrieval system to improve its performance by increasing the accuracy of the retrieved documents. This approach can help to reduce the ambiguity in text interpretation. The selection of actions involves querying and extracting new sources of information repetitively. Actions have two components, a reconciliation decision and a query choice. The reward is designed to maximize the extraction accuracy of the values, and at the same time the number of queries is minimized. The experimental work with two domains shows an improvement over traditional information extractors of 5% on average.

News feed recommendation can be seen as a combinatorial optimization problem and therefore it can be modeled as a MDP. He et al. (2016) studied the prediction of popular Reddit threads using a bi-directional LSTM architecture and RL. Another approach to the same problem involves the incorporation of global context available in the form of discussions from an external source of knowledge (He et al. 2017). An interesting idea explored in this approach is the use of two Q-functions. The first is used to generate a first ranking of the actions and the second one is utilized to rerank top action candidates. By doing this, good actions can be selected, i.e., These actions could otherwise be missed due to the very skewed action space that the algorithm can deal with.

Quite often we see that dialogue systems provide semantically correct responses which are not necessarily consistent with contextual facts. Mesgar et al. (2021) used RL to fine-tune the responses, optimizing for consistency and semantics.

Gao et al. (2019) approached another language processing task using RL, namely document summarization. The proposed paradigm uses learning-to-rank as a way to learn a reward function that is later used to generate near-optimal summaries.

8 Promising research directions

Based on our analysis of the problems and approaches here reported, we now take a step further and describe nine research directions that we believe will benefit from a RL approach in the coming years.

1. **Recognition of the user's input.** We noticed that a common element missing or at least underrepresented in NLP research is the recognition of the user's input. Commonly, this is treated as being inherently uncertain and most research accepts this and tries to cope with it without attempting to solve the source of the problems. This along with all other machine perception problems are very challenging tasks and far from being solved. We argue that trying to address uncertainty of the user input at the initial stages

would be more fruitful than simply regarding it as given. Thus, we argue that a future research direction would be to develop a reinforcement learning approach for generating internal semantic representations of the user's message from which other fields within and beyond NLP could benefit.

2. **Internal representation learning.** Learning an internal representation of language is a more general research direction. By using deep neural networks and reinforcement learning methods, it is possible to learn to code and decode sequences of text (Guo 2015). Although such an architecture was implemented and tested only with a text rephrasing task, we believe that the underlying problem of learning an internal representation of language is inherently related to some of the most important NLP problems, such as text understanding, MT, language generation, dialogue system management, parsing, etc. By solving the internal representation problem of language, we may partially solve the aforementioned problems to some extent. Therefore, research on deep learning and RL methods in a joint approach is currently of great importance to advance the state of the art in NLP systems.
3. **Exploitation of domain knowledge.** Another interesting research path is the one aiming at discovering ways to enhance RL through the exploitation of domain knowledge available in the form of natural language, as surveyed by Luketina et al. (2019). Some current trends involve methods studying knowledge transfer from descriptive task-dependent language corpora (Narasimhan et al. 2018). Pre-trained information retrieval systems can be integrated with RL agents (Chen et al. 2017) to improve the quality of the queries. Moreover, relevant information can be extracted from sources of unstructured data such as game manuals (Branavan et al. 2012).
4. **Exploitation of embodiment.** A trend in supervised language learning research considers the importance of embodiment for the emergence of language (Antunes et al. 2019; Heinrich et al. 2020). Multimodal inputs, such as an agent knowing its actuators while performing an action, help in classifying and verbally describing an action and allows better generalisation to novel action–object combinations (Eisermann et al. 2021). Embodied language learning has recently been brought to RL scenarios, specifically question answering where an agent needs to navigate in a scene to answer the questions (Tan and Liu 2020), or where it needs to perform actions on objects to answer questions (Deng et al. 2020). Like dialogue grounded in vision (Das et al. 2017), such interactive scenarios extend language learning into multiple modalities. Such applied scenarios also allow to introduce tasks, corresponding rewards, and hence seamless integration of language learning with RL. DRL neural architectures are a promising research path for the processing of multiple modalities in embodied language learning in a dynamic world.
5. **Language evolution.** From a more linguistic point of view, the study of language evolution using a RL perspective is also a fertile field for research. This process can be modelled by a multi-agent system, where a collection of agents is capable to create their own communication protocol by means of interaction with a common environment and by applying RL rules (Mordatch and Abbeel 2018). This kind of research can benefit from the recent advances in multi-agent systems and rising computational power. Moreover, research on cognitive robotics using neural models together with RL methods (Cruz et al. 2018; Cruz et al. 2018; Röder et al. 2020; Eppe et al. 2019; Hafez et al. 2019) has reached a point where the addition of language evolution capabilities seems to be more promising than ever before.
6. **Word embeddings.** More important, from our point of view, are the advances in neural language models, especially those for word embedding. The recent trend of continuous

language representations might have a huge potential if it is used together with RL. Word2vec (Mikolov et al. 2013) supplies a continuous vector representation of words. In a continuous BOWs architecture, Word2vec trains a simple neural network to predict a word from its surrounding words, achieving on its hidden layer a low-dimensional continuous representation of words in some semantically meaningful topology. Other word embeddings are GloVe (Pennington et al. 2014), which yields a similar performance more efficiently by using a co-occurrence matrix of words in their context, and FastText (Bojanowski et al. 2017), which includes subword information to enrich word vectors and to deal with out-of-vocabulary words.

A more powerful class of embeddings are contextualized word embeddings, which use the context, i.e., previous and following words, to embed a word. Two recent models are ELMo (Peters et al. 2018), which uses bidirectional LSTM, and BERT (Devlin et al. 2019), which uses a deep feedforward Transformer network architecture with self-attention. Both are character-based and hence, like FastText, use morphological cues and deal with out-of-vocabulary words. By taking into account the context, they handle different meanings of a word (e.g., "He touches a rock" vs. "He likes rock"). However, simple word embeddings become meaning embeddings, blurring the distinction between word- and sentence embeddings.

For the representations of utterances, Word2vec has been extended to Doc2vec (Le and Mikolov 2014), and other simple schemes are based on a weighted combination of contained word vectors (Arora et al. 2017; Rücklé et al. 2018). However, since these simple BOWs approaches lose word-order information, the original sentence cannot be reconstructed. Sentence generation is also difficult for supervised sentence embeddings such as InferSent (Conneau et al. 2017) or Google's Universal Sentence Encoder (Cer et al. 2018).

An unsupervised approach to sentence vectors are Skip-Thought Vectors (Kiros et al. 2015), which are trained to reconstruct the surrounding sentences of an encoded one. A simpler model would be an encoder–decoder autoencoder architecture, where the decoder reconstructs the same utterance that the encoder gets as input, based on a constant-length internal representation. Hence, this is a constant size continuous vector representation of an utterance, from which the utterance, which itself could consist of continuous word vectors, could also be reproduced.

To train utterance vectors on dialogues, large dialogue corpora exist, which can be classified into human–machine or human–human; spontaneously spoken, scripted spoken, or written (Serban et al. 2018). Examples are datasets of annotated telephone dialogues, movie dialogues, movie recommendation dialogues, negotiation dialogues, human–robot interaction, and also QA contains elements of dialogues.

Such continuous language representations could seamlessly play together with continuous RL algorithms like CACLA (van Hasselt and Wiering 2007), Deterministic Policy Gradient (DPG, Silver et al. 2014) or deep DPG (DDPG, Lillicrap et al. 2015). These algorithms handle continuous state input and continuous action output. Actions of a dialogue agent would be the agent's utterances, which would result in a new state after the response of its communication partner. Continuous utterance representations would allow optimization of an action by gradient ascent to maximize certain rewards which express desired future state properties. For example, it could be desired to maximize the positive sentiment of an upcoming utterance which can be estimated by a differentiable neural network (Bothe et al. 2017).

Other possible desired state properties could be to maximize a human's excitement in order to motivate him to make a decision; to maximize the duration of the conversation,

or lead it to an early end with a pleased human; to acquire certain information from, or to pass on information to the human. However, not all goals can be easily expressed as points in a continuous utterance space that represents a dialogue. To this end, future research on language needs to be extended towards representing more of its semantics, which entails understanding the entire situation.

7. **Intelligent conversational systems.** When conversing with chatbots, it is common to end up in the situation where the bot starts responding with "I don't know what you are talking about" repeatedly, no matter what it is asked. This problem is identified as the generic response problem. The cause for this problem might be that such kind of answers occur very often in the training set. Also they are highly compatible with various questions (Li et al. 2016). Another issue is when a dataset has similar responses to different contexts (Sankar and Ravi 2019). One way to improve the efficiency in RL is through the combination of model-based and model-free learning (Hafez et al. 2020). We propose that this approach might be useful to solve the generic response problem.

   Furthermore, all the experience gained from working with algorithms designed for text-based games and applications on learning of navigational directions can be extended and adapted to be useful in the implementation of intelligent tutors, smart enough to understand the questions posed by the user and select the most appropriate learning resource, whether it is some text, audio, video, hyperlink, etc. Those intelligent tutors can improve over time.
8. **Assessment of conversational systems.** Finally, in conversational systems, a critical point that needs further investigation is the definition of robust evaluation schemes that can be automated and used to assess the quality of automatic dialogue systems. Currently, the performance of such systems is measured through ad hoc procedures that depend on the specific application and most importantly, they require the intervention of a human, which makes these systems very difficult to be scaled.
9. **Document-editing RL Assistants.**Kudashkina et al. (2020) proposed the domain of voice document editing as a particularly well-suited one for the development of RL intelligent assistants that can engage in a conversation. They argue that in voice document editing, the domain is clearly defined, delimited and the agent has full access to it. These conditions are advantageous for an agent that learns the domain of discourse through model-based RL. Important future research questions the authors mention are, first, what level of ambition should the agent's learning have? And second, how should the training of the assistant be performed, online or offline?

9 Conclusions

We have provided a review of the main categories of NLP problems that have been approached using reinforcement learning methods. Some of these problems considered reinforcement learning as the main algorithm, such as the dialogue management systems. In others, RL was used marginally, only to partially help in the solution of the central problem. In both cases, RL algorithms have played an important part in the optimization of control policies through the self-exploration of the states and actions.

With the current advances in RL algorithms, especially with those algorithms in which the value functions and policy functions are replaced with deep neural networks, it is impossible not to consider that RL will play a major role in solving some of the most important NLP problems. Especially, we have witnessed solid evidence that algorithms

with self-improvement and self-adaptation capabilities have pushed the performance in challenging machine learning problems to the next level.

Currently, none of the NLP tasks here analyzed have RL methods as state-of-the-art methodologies. Many of the problems are being solved with increasing success using transformer neural network models such as BERT and GPT. However, we argue that RL can be jointly applied with deep neural models. RL can provide benefit by its inherent exploratory capacity. This is, reinforcement learning can help find better actions and better states due to its credit assignment approach. The best policies found by neural networks, such as transformers, can potentially get fine-tuned by reinforcements.

Acknowledgements This work received partial support from the German Research Foundation (DFG) under projects CML (TRR-169) and LeCAREbot, and from the Federal Ministry for Economic Affairs and Climate Action (BMWK) under project SIDIMO. We thank Burhan Hafez for discussions and providing references highly relevant to this review.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

Antunes A, Laflaquiere A, Ogata T, Cangelosi A (2019) A bi-directional multiple timescales LSTM model for grounding of actions and verbs. In: IEEE/RSJ international conference on intelligent robots and systems (IROS), Macau, China, pp 2614–2621

Arora S, Liang Y, Ma T (2017) A simple but tough-to-beat baseline for sentence embeddings. In: International conference on learning representations (ICLR), Toulon, France. OpenReview.net

Bahdanau D, Cho K, Bengio Y (2015) Neural machine translation by jointly learning to align and translate. In: International conference on learning representations (ICLR), San Diego, CA, USA. arxiv

Bengio S, Vinyals O, Jaitly N, Shazeer N (2015) Scheduled sampling for sequence prediction with recurrent neural networks. In: International conference on neural information processing systems (NIPS), Montreal, QC, Canada, vol 1. MIT Press, pp 1171–1179

Bojanowski P, Grave E, Joulin A, Mikolov T (2017) Enriching word vectors with subword information. Trans Assoc Comput Linguist 5:135–146

Bothe C, Magg S, Weber C, Wermter S (2017) Dialogue-based neural learning to estimate the sentiment of a next upcoming utterance. In: Lintas A, Rovetta S, Verschure PF, Villa AE (eds) International conference on artificial neural networks (ICANN), Alghero, Italy. Lecture notes in computer science, vol 10614. Springer, pp 477–485

Branavan SRK, Silver D, Barzilay R (2012) Learning to win by reading manuals in a Monte Carlo framework. J Artif Intell Res 43:661–704

Brown PF, Cocke J, Pietra SAD, Pietra VJD, Jelinek F, Lafferty JD, Mercer RL, Roossin PS (1990) A statistical approach to machine translation. Comput Linguist 16(2):79–85

Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S, Herbert-Voss A, Krueger G, Henighan T, Child R, Ramesh A, Ziegler DM, Wu J, Winter C, Hesse C, Chen M, Sigler E, Litwin M, Gray S, Chess B, Clark J, Berner C, McCandlish S, Radford A, Sutskever I, Amodei D (2020) Language models are few-shot learners. In: Neural information processing systems (NeurIPS). Online conference

Cangelosi A, Parisi D (eds) (2002) Simulating the evolution of language. Springer, London

Cao R, Zhu S, Liu C, Li J, Yu K (2019) Semantic parsing with dual learning. In: Annual meeting of the Association for Computational Linguistics (ACL), Florence, Italy, vol 57. Association for Computational Linguistics, pp 51–64.https://doi.org/10.18653/v1/P19-1007

Cer D, Yang Y, Kong Sy, Hua N, Limtiaco N, John RS, Constant N, Guajardo-Cespedes M, Yuan S, Tar C, Sung YH, Strope B, Kurzweil R (2018) Universal sentence encoder. arXiv:1803.11175 [cs]

Che T, Li Y, Zhang R, Hjelm RD, Li W, Song Y, Bengio Y (2017) Maximum-likelihood augmented discrete generative adversarial networks. arXiv:1702.07983 [cs]

Chen D, Fisch A, Weston J, Bordes A (2017) Reading Wikipedia to answer open-domain questions. In: Annual meeting of the Association for Computational Linguistics (ACL), Vancouver, BC, Canada, vol 55. Association for Computational Linguistics, pp. 1870–1879. https://doi.org/10.18653/v1/P17-1171

Chen L, Yang R, Chang C, Ye Z, Zhou X, Yu K (2017) On-line dialogue policy learning with companion teaching. In: Conference of the European Chapter of the Association for Computational Linguistics (EACL), Valencia, Spain. Short papers, vol 15. Association for Computational Linguistics, pp 198–204

Chen L, Zhou X, Chang C, Yang R, Yu K (2017) Agent-aware dropout DQN for safe and efficient on-line dialogue policy learning. In: Conference on empirical methods in natural language processing (EMNLP), Copenhagen, Denmark. Association for Computational Linguistics, pp 2454–2464. https://doi.org/10.18653/v1/D17-1260

Chen Z, Chen L, Liu X, Yu K (2020) Distributed structured actor-critic reinforcement learning for universal dialogue management. IEEE/ACM Trans Audio Speech Lang Process 28:2400–2411. https://doi.org/10.1109/TASLP.2020.3013392

Cho K, van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: Conference on empirical methods in natural language processing (EMNLP), Doha, Qatar. Association for Computational Linguistics, pp 1724–1734. https://doi.org/10.3115/v1/D14-1179

Chomsky N (1959) On certain formal properties of grammars. Inf Control 2(2):137–167. https://doi.org/10.1016/S0019-9958(59)90362-6

Chomsky N (1965) Aspects of the theory of syntax. The MIT Press, Cambridge

Conneau A, Kiela D, Schwenk H, Barrault L, Bordes A (2017) Supervised learning of universal sentence representations from natural language inference data. In: Conference on empirical methods in natural language processing (EMNLP), Copenhagen, Denmark. Association for Computational Linguistics, pp 670–680. https://doi.org/10.18653/v1/D17-1070

Crook PA, Keizer S, Wang Z, Tang W, Lemon O (2014) Real user evaluation of a POMDP spoken dialogue system using automatic belief compression. Comput Speech Lang 28(4):873–887. https://doi.org/10.1016/j.csl.2013.12.002

Cruz F, Magg S, Nagai Y, Wermter S (2018) Improving interactive reinforcement learning: what makes a good teacher? Connect Sci 30(3):306–325. https://doi.org/10.1080/09540091.2018.1443318

Cruz F, Parisi GI, Wermter S (2018) Multi-modal feedback for affordance-driven interactive reinforcement learning. In: International joint conference on neural networks (IJCNN), Rio de Janeiro, Brazil, pp 1–8. https://doi.org/10.1109/IJCNN.2018.8489237

Cuayáhuitl H, Kruijff-Korbayová I, Dethlefs N (2014) Nonstrict hierarchical reinforcement learning for interactive systems and robots. ACM Trans Interact Intell Syst 4(3):15:1-15:30. https://doi.org/10.1145/2659003

Das A, Kottur S, Moura JMF, Lee S, Batra D (2017) Learning cooperative visual dialog agents with deep reinforcement learning. In: IEEE international conference on computer vision (ICCV), Venice, Italy, pp 2951–2960. https://doi.org/10.1109/ICCV.2017.321

Das R, Dhuliawala S, Zaheer M, Vilnis L, Durugkar I, Krishnamurthy A, Smola A, McCallum A (2018) Go for a walk and arrive at the answer: reasoning over paths in knowledge bases using reinforcement learning. In: International conference on learning representations (ICLR), Vancouver, BC, Canada

Daumé H III, Langford J, Marcu D (2009) Search-based structured prediction. Mach Learn 75(3):297–325. https://doi.org/10.1007/s10994-009-5106-x

Deng Y, Guo X, Zhang N, Guo D, Liu H, Sun F (2020) MQA: answering the question via robotic manipulation. arXiv:2003.04641 [cs]

Dethlefs N, Cuayáhuitl H (2011) Combining hierarchical reinforcement learning and Bayesian networks for natural language generation in situated dialogue. In: European workshop on natural language generation (ENLG), Nancy, France, vol 11. Association for Computational Linguistics, pp 110–120

Dethlefs N, Cuayáhuitl H (2011) Hierarchical reinforcement learning and hidden Markov models for task-oriented natural language generation. In: Annual meeting of the Association for

Computational Linguistics: human language technologies (ACL). Short papers, Portland, OR, USA, vol 49. Association for Computational Linguistics, pp 654–659

Devlin J, Chang MW, Lee K, Toutanova K (2019) BERT: pre-training of deep bidirectional transformers for language understanding. In: Conference of the North American Chapter of the Association for Computational Linguistics: human language technologies (NAACL HLT), Minneapolis, MN, USA. Association for Computational Linguistics, pp. 4171–4186. https://doi.org/10.18653/v1/N19-1423

Devlin J, Zbib R, Huang Z, Lamar T, Schwartz R, Makhoul J (2014) Fast and robust neural network joint models for statistical machine translation. In: Annual meeting of the Association for Computational Linguistics (ACL), Baltimore, MD, USA, vol 52. Association for Computational Linguistics, pp 1370–1380. https://doi.org/10.3115/v1/P14-1129

Eisermann A, Lee JH, Weber C, Wermter S (2021) Generalization in multimodal language learning from simulation. In: International joint conference on neural networks (IJCNN), Shenzhen, China. pp 1–8. https://doi.org/10.1109/IJCNN52387.2021.9534275

Eppe M, Nguyen PDH, Wermter S (2019) From semantics to execution: integrating action planning with reinforcement learning for robotic causal problem-solving. Front Robot AI. https://doi.org/10.3389/frobt.2019.00123

Fügen C, Waibel A, Kolss M (2007) Simultaneous translation of lectures and speeches. Mach Transl 21(4):209–252. https://doi.org/10.1007/s10590-008-9047-0

Gao J, Galley M, Li L (2018) Neural approaches to conversational AI. In: International ACM SIGIR conference on research and development in information retrieval, Ann Arbor, MI, USA, vol 41. Association for Computing Machinery, pp 1371–1374

Gao Y, Meyer C, Mesgar M, Gurevych I (2019) Reward learning for efficient reinforcement learning in extractive document summarisation. In: 19th International joint conference on artificial intelligence (IJCAI), Macao, China. AAAI Press, pp 2350–2356

Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Advances in neural information processing systems (NIPS), Montreal, QC, Canada, vol 27. Curran Associates, Inc., pp 2672–2680

Grissom II A, He H, Boyd-Graber J, Morgan J, Daumé III H (2014) Don't until the final verb wait: reinforcement learning for simultaneous machine translation. In: Conference on empirical methods in natural language processing (EMNLP), Doha, Qatar. Association for Computational Linguistics, pp 1342–1352. https://doi.org/10.3115/v1/D14-1140

Gu J, Neubig G, Cho K, Li VO (2017) Learning to translate in real-time with neural machine translation. In: Conference of the European Chapter of the Association for Computational Linguistics (EACL), Valencia, Spain, vol 15. Association for Computational Linguistics, pp 1053–1062

Guo H (2015) Generating text with deep reinforcement learning. In: NIPS deep reinforcement learning workshop, Montreal, QC, Canada

Guo J, Lu S, Cai H, Zhang W, Yu Y, Wang J (2018) Long text generation via adversarial training with leaked information. Proc AAAI Conf Artif Intell 32(1):5141–5148

Guo X, Klinger T, Rosenbaum C, Bigus JP, Campbell M, Kawas B, Talamadupula K, Tesauro G, Singh S (2017) Learning to query, reason, and answer questions on ambiguous texts. In: International conference on learning representations (ICLR), Toulon, France

Hafez MB, Weber C, Kerzel M, Wermter S (2019) Deep intrinsically motivated continuous actor-critic for efficient robotic visuomotor skill learning. Paladyn J Behav Robot 10(1):14–29. https://doi.org/10.1515/pjbr-2019-0005

Hafez MB, Weber C, Kerzel M, Wermter S (2020) Improving robot dual-system motor learning with intrinsically motivated meta-control and latent-space experience imagination. Robot Auton Syst 133:103630

Hassan H, Aue A, Chen C, Chowdhary V, Clark J, Federmann C, Huang X, Junczys-Dowmunt M, Lewis W, Li M, Liu S, Liu TY, Luo R, Menezes A, Qin T, Seide F, Tan X, Tian F, Wu L, Wu S, Xia Y, Zhang D, Zhang Z, Zhou M (2018) Achieving human parity on automatic Chinese to English news translation. arXiv:1803.05567 [cs]

He D, Lu H, Xia Y, Qin T, Wang L, Liu TY (2017) Decoding with value networks for neural machine translation. In: International conference on neural information processing systems (NIPS), Long Beach, CA, USA, vol 30. Curran Associates, Inc., pp 177–186

He D, Xia Y, Qin T, Wang L, Yu N, Liu TY, Ma WY (2016) Dual learning for machine translation. In: Advances in neural information processing systems (NIPS), Barcelona, Spain, vol 29, pp 820–828

He J, Chen J, He X, Gao J, Li L, Deng L, Ostendorf M (2016) Deep reinforcement learning with a natural language action space. In: Annual meeting of the Association for Computational Linguistics (ACL), Berlin, Germany, vol 54. Association for Computational Linguistics, pp 1621–1630

He J, Ostendorf M, He X (2017) Reinforcement learning with external knowledge and two-stage Q-functions for predicting popular Reddit threads. arXiv:1704.06217 [cs]

He J, Ostendorf M, He X, Chen J, Gao J, Li L, Deng L (2016) Deep reinforcement learning with a combinatorial action space for predicting popular Reddit threads. In: Conference on empirical methods in natural language processing (EMNLP), Austin, TX, USA. Association for Computational Linguistics, pp 1838–1848. https://doi.org/10.18653/v1/D16-1189

Heinrich S, Yao Y, Hinz T, Liu Z, Hummel T, Kerzel M, Weber C, Wermter S (2020) Crossmodal language grounding in an embodied neurocognitive model. Front Neurorobot. https://doi.org/10.3389/fnbot.2020.00052

Henderson J, Lemon O, Georgila K (2008) Hybrid reinforcement/supervised learning of dialogue policies from fixed datasets. Comput Linguist 34(4):487–511

Higashinaka R, Mizukami M, Funakoshi K, Araki M, Tsukahara H, Kobayashi Y (2015) Fatal or not? Finding errors that lead to dialogue breakdowns in chat-oriented dialogue systems. In: Conference on empirical methods in natural language processing (EMNLP), Lisbon, Portugal. Association for Computational Linguistics, pp 2243–2248

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hutchins WJ, Somers HL (1992) An introduction to machine translation. Academic, London

Jiang J, Teichert A, Eisner J, Daumé III H (2012) Learned prioritization for trading off accuracy and speed. In: Advances in neural information processing systems (NIPS), Lake Tahoe, NV, USA, vol 25

Jurcicek F, Thomson B, Keizer S, Mairesse F, Gasic M, Yu K, Young SJ (2010) Natural belief-critic: a reinforcement algorithm for parameter estimation in statistical spoken dialogue systems. In: Annual conference of the International Speech Communication Association (INTERSPEECH), Makuhari, Japan, pp 90–93

Kalchbrenner N, Blunsom P (2013) Recurrent continuous translation models. In: Conference on empirical methods in natural language processing (EMNLP), Seattle, WA, USA. Association for Computational Linguistics, pp 1700–1709

Keneshloo Y, Shi T, Ramakrishnan N, Reddy CK (2020) Deep reinforcement learning for sequence-to-sequence models. IEEE Trans Neural Netw Learn Syst 31(7):2469–2489. https://doi.org/10.1109/TNNLS.2019.2929141

Kiros R, Zhu Y, Salakhutdinov RR, Zemel R, Urtasun R, Torralba A, Fidler S (2015) Skip-thought vectors. In: Advances in neural information processing systems (NIPS), Montreal, QC, Canada, vol 28. Curran Associates, Inc., pp 3294–3302

Koehn P (2009) Statistical machine translation. Cambridge University Press, Cambridge

Koehn P, Och FJ, Marcu D (2003) Statistical phrase-based translation. In: Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL), Edmonton, AB, Canada. Association for Computational Linguistics, pp 48–54. https://doi.org/10.3115/1073445.1073462

Kübler S, McDonald R, Nivre J (2008) Dependency Parsing. Synth Lect Hum Lang Technol 2(1):1–127. https://doi.org/10.2200/S00169ED1V01Y200901HLT002

Kudashkina K, Pilarski PM, Sutton RS (2020) Document-editing assistants and model-based reinforcement learning as a path to conversational AI. arXiv:2008.12095 [cs]

Lam TK, Schamoni S, Riezler S (2019) Interactive–predictive neural machine translation through reinforcement and imitation. In: Proceedings of machine translation summit XVII: research track, Dublin, Ireland, vol 1. European Association for Machine Translation, pp 96–106

Langford J, Zhang T (2007) The epoch-greedy algorithm for contextual multi-armed bandits. In: Advances in neural information processing systems (NIPS), 2007, Vancouver, BC, Canada, vol 20. Curran Associates, Inc., pp 817–824

Lê M, Fokkens A (2017) Tackling error propagation through reinforcement learning: a case of greedy dependency parsing. In: Conference of the European Chapter of the Association for Computational Linguistics (EACL), Valencia, Spain, vol 1. Association for Computational Linguistics, pp 677–687

Le Q, Mikolov T (2014) Distributed representations of sentences and documents. In: International conference on machine learning (ICML), Beijing, China, vol 32. PMLR, pp 1188–1196

LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444. https://doi.org/10.1038/nature14539

Lemon O (2011) Learning what to say and how to say it: joint optimisation of spoken dialogue management and natural language generation. Comput Speech Lang 25(2):210–221. https://doi.org/10.1016/j.csl.2010.04.005

Levin E, Pieraccini R, Eckert W (2000) A stochastic model of human–machine interaction for learning dialog strategies. IEEE Trans Speech Audio Process 8(1):11–23. https://doi.org/10.1109/89.817450

Li J, Monroe W, Ritter A, Galley M, Gao J, Jurafsky D (2016) Deep reinforcement learning for dialogue generation. In: Conference on empirical methods in natural language processing (EMNLP), Austin, TX, USA. Association for Computational Linguistics, pp 1192–1202

Li L, Chu W, Langford J, Schapire RE (2010) A contextual-bandit approach to personalized news article recommendation. In: International conference on world wide web (WWW), Raleigh, NC, USA, vol 19. Association for Computing Machinery, pp 661–670. https://doi.org/10.1145/1772690.1772758

Li X, Chen YN, Li L, Gao J, Celikyilmaz A (2017) End-to-end task-completion neural dialogue systems. In: International joint conference on natural language processing (IJCNLP), Taipei, Taiwan. Asian Federation of Natural Language Processing, pp 733–743

Li X, Lipton ZC, Dhingra B, Li L, Gao J, Chen YN (2017) A user simulator for task-completion dialogues. arXiv:1612.05688 [cs]

Li Z, Jiang X, Shang L, Li H (2018) Paraphrase generation with deep reinforcement learning. In: Conference on empirical methods in natural language processing (EMNLP), Brussels, Belgium. Association for Computational Linguistics, pp 3865–3878. https://doi.org/10.18653/v1/D18-1421

Lillicrap TP, Hunt JJ, Pritzel A, Heess N, Erez T, Tassa Y, Silver D, Wierstra D (2015) Continuous control with deep reinforcement learning. arXiv:1509.02971

Lin K, Li D, He X, Zhang Z, Sun Mt (2017) Adversarial ranking for language generation. In: Advances in neural information processing systems (NIPS), Long Beach, CA, USA, vol 30. Curran Associates, Inc.

Litman DJ, Kearns MS, Singh SP, Walker MA (2000) Automatic optimization of dialogue management. In: International conference on computational linguistics (COLING), vol 18, Saarbrücken, Germany. Association for Computational Linguistics, pp 502–508

Liu Q, Chen Y, Chen B, Lou JG, Chen Z, Zhou B, Zhang D (2020) You impress me: dialogue generation via mutual persona perception. In: Annual meeting of the Association for Computational Linguistics (ACL), vol 58. Association for Computational Linguistics, pp 1417–1427.https://doi.org/10.18653/v1/2020.acl-main.131

Lu K, Zhang S, Chen X (2019) Goal-oriented dialogue policy learning from failures. Proc AAAI Conf Artif Intell 33(01):2596–2603

Luketina J, Nardelli N, Farquhar G, Foerster J, Andreas J, Grefenstette E, Whiteson S, Rocktäschel T (2019) A survey of reinforcement learning informed by natural language. In: 28th International joint conference on artificial intelligence (IJCAI), Macau, China, pp 6309–6317.https://doi.org/10.24963/ijcai.2019/880

Mesgar M, Simpson E, Gurevych I (2021) Improving factual consistency between a response and persona facts. In: Conference of the European Chapter of the Association for Computational Linguistics (EACL), Main Volume. Association for Computational Linguistics, pp 549–562. https://doi.org/10.18653/v1/2021.eacl-main.44

Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv:1301.3781 [cs]

Mnih V, Badia AP, Mirza M, Graves A, Lillicrap T, Harley T, Silver D, Kavukcuoglu K (2016) Asynchronous methods for deep reinforcement learning. In: 33rd International conference on machine learning (ICML), proceedings of machine learning research (PMLR), New York, NY, USA, vol 48, pp 1928–1937

Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, Petersen S, Beattie C, Sadik A, Antonoglou I, King H, Kumaran D, Wierstra D, Legg S, Hassabis D (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533

Mordatch I, Abbeel P (2018) Emergence of grounded compositional language in multi-agent populations. In: Proceedings of the AAAI conference on artificial intelligence, vol 32(1)

Narasimhan K, Barzilay R, Jaakkola T (2018) Grounding language for transfer in deep reinforcement learning. J Artif Intell Res 63:849–874

Narasimhan K, Kulkarni TD, Barzilay R (2015) Language understanding for text-based games using deep reinforcement learning. In: Conference on empirical methods for natural language processing (EMNLP), Lisbon, Portugal. Association for Computational Linguistics, pp 1–11

Narasimhan K, Yala A, Barzilay R (2016) Improving information extraction by acquiring external evidence with reinforcement learning. In: Conference on empirical methods in natural language processing (EMNLP), Austin, TX, USA. Association for Computational Linguistics, pp 2355–2365. https://doi.org/10.18653/v1/D16-1261

Neu G, Szepesvári C (2009) Training parsers by inverse reinforcement learning. Mach Learn 77(2):303. https://doi.org/10.1007/s10994-009-5110-1

Ng AY, Russell SJ (2000) Algorithms for inverse reinforcement learning. In: International conference on machine learning (ICML), Stanford, CA, USA, vol 17. Morgan Kaufmann Publishers, Inc., pp 663–670

Och FJ (2003) Minimum error rate training in statistical machine translation. In: 41st Annual meeting on Association for Computational Linguistics (ACL), Sapporo, Japan, vol 1. Association for Computational Linguistics, pp 160–167

Papaioannou I, Lemon O (2017) Combining chat and task-based multimodal dialogue for more engaging HRI: a scalable method using reinforcement learning. In: ACM/IEEE international conference on human–robot interaction (HRI), Vienna, Austria. ACM, pp. 365–366. https://doi.org/10.1145/3029798.3034820

Papangelis A, Namazifar M, Khatri C, Wang YC, Molino P, Tur G (2020) Plato dialogue system: a flexible conversational AI research platform. arXiv:2001.06463 [cs]

Papangelis A, Wang YC, Molino P, Tur G (2019) Collaborative multi-agent dialogue model training via reinforcement learning. In: Annual SIGdial meeting on discourse and dialogue (SIGDIAL), Stockholm, Sweden, vol 20. Association for Computational Linguistics, pp. 92–102. https://doi.org/10.18653/v1/W19-5912

Papineni K, Roukos S, Ward T, Zhu WJ (2002) BLEU: a method for automatic evaluation of machine translation. In: Annual meeting of the Association for Computational Linguistics (ACL), Philadelphia, Pennsylvania, USA, vol 40. Association for Computational Linguistics, pp. 311–318. https://doi.org/10.3115/1073083.1073135

Pennington J, Socher R, Manning C (2014) GloVe: global vectors for word representation. In: Conference on empirical methods in natural language processing (EMNLP), Doha, Qatar. Association for Computational Linguistics, pp 1532–1543. https://doi.org/10.3115/v1/D14-1162

Peters M, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, Zettlemoyer L (2018) Deep contextualized word representations. In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), New Orleans, LA, USA. Association for Computational Linguistics, pp 2227–2237

Poljak BT (1973) Pseudogradient adaptation and training algorithms. Avtom Telemeh 3:45–68

Röder F, Eppe M, Nguyen PDH, Wermter S (2020) Curious hierarchical actor-critic reinforcement learning. In: International conference on artificial neural networks (ICANN). Lecture notes in computer science, Bratislava, Slovakia. Springer, pp 408–419

Rücklé A, Eger S, Peyrard M, Gurevych I (2018) Concatenated power mean word embeddings as universal cross-lingual sentence representations. arXiv:1803.01400 [cs]

Russell S, Norvig P (2010) Artificial intelligence: a modern approach, 3rd edn. Pearson, Harlow

Sankar C, Ravi S (2019) Deep reinforcement learning for modeling chit-chat dialog with discrete attributes. In: Annual SIGdial meeting on discourse and dialogue, Stockholm, Sweden, vol 20. Association for Computational Linguistics, pp 1–10

Schatzmann J, Young S (2009) The hidden agenda user simulation model. IEEE Trans Audio Speech Lang Process 17(4):733–747. https://doi.org/10.1109/TASL.2008.2012071

Schrittwieser J, Antonoglou I, Hubert T, Simonyan K, Sifre L, Schmitt S, Guez A, Lockhart E, Hassabis D, Graepel T, Lillicrap T, Silver D (2020) Mastering Atari, Go, Chess and Shogi by planning with a learned model. Nature 588(7839):604–609

Schulman J, Levine S, Abbeel P, Jordan M, Moritz P (2015) Trust region policy optimization. In: International conference on machine learning (ICML), proceedings of machine learning research (PMLR), Lille, France, vol 37, pp 1889–1897

Serban IV, Lowe R, Henderson P, Charlin L, Pineau J (2018) A survey of available corpora for building data-driven dialogue systems: the journal version. Dialogue Discourse 9(1):1–49. https://doi.org/10.5087/dad.2018.101

Shi Z, Chen X, Qiu X, Huang X (2018) Toward diverse text generation with inverse reinforcement learning. In: International joint conference on artificial intelligence (IJCAI), Stockholm, Sweden, vol 27, pp 4361–4367. https://doi.org/10.24963/ijcai.2018/606

Shum HY, He XD, Li D (2018) From Eliza to XiaoIce: challenges and opportunities with social chatbots. Front Inf Technol Electron Eng 19(1):10–26. https://doi.org/10.1631/FITEE.1700826

Silver D, Huang A, Maddison CJ, Guez A, Sifre L, van den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lillicrap T, Leach M, Kavukcuoglu K, Graepel T, Hassabis D (2016) Mastering the game of go with deep neural networks and tree search. Nature 529(7587):484–489. https://doi.org/10.1038/nature16961

Silver D, Lever G, Heess N, Degris T, Wierstra D, Riedmiller M (2014) Deterministic policy gradient algorithms. In: 31st International conference on machine learning (ICML). Proceedings of machine learning research (PMLR), Beijing, China, vol 32, pp 387–395

Silver D, Schrittwieser J, Simonyan K, Antonoglou I, Huang A, Guez A, Hubert T, Baker L, Lai M, Bolton A, Chen Y, Lillicrap T, Hui F, Sifre L, van den Driessche G, Graepel T, Hassabis D (2017) Mastering the game of go without human knowledge. Nature 550(7676):354–359. https://doi.org/10.1038/nature24270

Singh S, Kearns M, Litman DJ, Walker MA (2000) Empirical evaluation of a reinforcement learning spoken dialogue system. In: National conference on artificial intelligence (AAAI), Austin, TX, USA, vol 17. AAAI Press, pp 645–651

Singh SP, Litman D, Kearns M, Walker M (2002) Optimizing dialogue management with reinforcement learning: experiments with the NJFun system. J Artif Intell Res 16:105–133. https://doi.org/10.1613/jair.859

Sipser M (2013) Introduction to the theory of computation, 3rd edn. Course technology. Cengage Learning, Boston

Sokolov A, Kreutzer J, Lo C, Riezler S (2016) Learning structured predictors from bandit feedback for interactive NLP. In: Annual meeting of the Association for Computational Linguistics (ACL), Berlin, Germany, vol 54. Association for Computational Linguistics, pp 1610–1620. https://doi.org/10.18653/v1/P16-1152

Sokolov A, Riezler S, Urvoy T (2015) Bandit structured prediction for learning from partial feedback in statistical machine translation. In: Proceedings of MT summit XV, Miami, FL, USA. Association for Machine Translation in the Americas, pp 160–171

Stahlberg F (2020) Neural machine translation: a review. J Artif Intell Res 69:343–418. https://doi.org/10.1613/jair.1.12007

Su PH, Gašić M, Young S (2018) Reward estimation for dialogue policy optimisation. Comput Speech Lang 51:24–43. https://doi.org/10.1016/j.csl.2018.02.003

Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: Advances in neural information processing systems (NIPS), Montreal, QC, Canada, vol 27. Curran Associates, Inc., pp 3104–3112

Sutton RS, Barto AG (2018) Reinforcement learning: an introduction, 2nd edn. Adaptive computation and machine learning series. The MIT Press, Cambridge

Tamar A, WU Y, Thomas G, Levine S, Abbeel P (2016) Value iteration networks. In: Advances in neural information processing systems (NIPS), Barcelona, Spain, vol 29. Curran Associates, Inc., pp. 2154–2162

Tan S, Liu H (2020) Towards embodied scene description. In: Robotics: science and systems. RSS Foundation, Corvallis

Thomson B, Young S (2010) Bayesian update of dialogue state: a POMDP framework for spoken dialogue systems. Comput Speech Lang 24(4):562–588

Ultes S, Rojas-Barahona LM, Su PH, Vandyke D, Kim D, Casanueva I, Budzianowski P, Mrkšić N, Wen TH, Gašić M, Young S (2017) PyDial: a multi-domain statistical dialogue system toolkit. In: Proceedings of system demonstrations, Vancouver, BC, Canada, vol 55. Association for Computational Linguistics, pp 73–78

van Hasselt H, Wiering MA (2007) Reinforcement learning in continuous action spaces. In: IEEE symposium on approximate dynamic programming and reinforcement learning (ADPRL), Honolulu, HI, USA, pp 272–279. https://doi.org/10.1109/ADPRL.2007.368199

Vogel A, Jurafsky D (2010) Learning to follow navigational directions. In: Annual meeting of the Association for Computational Linguistics (ACL), Uppsala, Sweden, vol 48. Association for Computational Linguistics, pp 806–814

Walker MA (2000) An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email. J Artif Intell Res 12:387–416. https://doi.org/10.1613/jair.713

Watkins CJCH (1989) Learning from delayed rewards. Dissertation, Cambridge University

Way A (2018) Quality expectations of machine translation. In: Moorkens J, Castilho S, Gaspari F, Doherty S (eds) Translation quality assessment: from principles to practice, machine translation: technologies and applications, vol 1. Springer, Cham, pp 159–178. https://doi.org/10.1007/978-3-319-91241-7_8

Weaver W (1955) Translation. In: Locke WN, Booth AD (eds) Machine translation of languages: fourteen essays. The MIT Press, Cambridge, pp 15–23

Williams JD, Young S (2007) Partially observable Markov decision processes for spoken dialog systems. Comput Speech Lang 21(2):393–422

Williams P, Sennrich R, Post M, Koehn P (2016) Syntax-based statistical machine translation, synthesis lectures on human language technologies, vol 9. Morgan & Claypool Publishers. https://doi.org/10.2200/S00716ED1V04Y201604HLT033

Wu L, Tian F, Qin T, Lai J, Liu TY (2018) A study of reinforcement learning for neural machine translation. In: Conference on empirical methods in natural language processing (EMNLP), Brussels, Belgium. Association for Computational Linguistics, pp 3612–3621. https://doi.org/10.18653/v1/D18-1397

Wu Y, Schuster M, Chen Z, Le QV, Norouzi M, Macherey W, Krikun M, Cao Y, Gao Q, Macherey K, Klingner J, Shah A, Johnson M, Liu X, Kaiser Ł, Gouws S, Kato Y, Kudo T, Kazawa H, Stevens K, Kurian G, Patil N, Wang W, Young C, Smith J, Riesa J, Rudnick A, Vinyals O, Corrado G, Hughes M, Dean J (2016) Google's neural machine translation system: bridging the gap between human and machine translation. Computing Research Repository (CoRR) in arXiv abs/1609.08144, 23

Wuebker J, Muehr S, Lehnen P, Peitz S, Ney H (2015) A comparison of update strategies for large-scale maximum expected BLEU training. In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT), Denver, CO, USA. Association for Computational Linguistics, pp 1516–1526. https://doi.org/10.3115/v1/N15-1175

Xiong W, Hoang T, Wang WY (2017) DeepPath: a reinforcement learning method for knowledge graph reasoning. In: Conference on empirical methods in natural language processing (EMNLP), Copenhagen, Denmark. Association for Computational Linguistics, pp 564–573. https://doi.org/10.18653/v1/D17-1060

Yang M, Huang W, Tu W, Qu Q, Shen Y, Lei K (2021) Multitask learning and reinforcement learning for personalized dialog generation: an empirical study. IEEE Trans Neural Netw Learn Syst 32(1):49–62

Young S, Gašić M, Keizer S, Mairesse F, Schatzmann J, Thomson B, Yu K (2010) The hidden information state model: a practical framework for POMDP-based spoken dialogue management. Comput Speech Lang 24(2):150–174

Young S, Gašić M, Thomson B, Williams JD (2013) POMDP-based statistical spoken dialog systems: a review. Proc IEEE 101(5):1160–1179

Young SJ (2000) Probabilistic methods in spoken-dialogue systems. Philos Trans Math Phys Eng Sci 358(1769):1389–1402

Yu L, Zhang W, Wang J, Yu Y (2017) SeqGAN: sequence generative adversarial nets with policy gradient. Proc AAAI Conf Artif Intell 31(1):2852–2858

Yu Z, Rudnicky A, Black A (2017) Learning conversational systems that interleave task and non-task content. In: International joint conference on artificial intelligence (IJCAI), Melbourne, VIC, Australia, vol 26, pp 4214–4220. https://doi.org/10.24963/ijcai.2017/589

Zhang L, Chan KP (2009) Dependency parsing with energy-based reinforcement learning. In: International conference on parsing technologies (IWPT), Paris, France, vol 11. Association for Computational Linguistics, pp 234–237

Zhao T, Xie K, Eskenazi M (2019) Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT), Minneapolis, Minnesota, vol 1. Association for Computational Linguistics, pp 1208–1218. https://doi.org/10.18653/v1/N19-1123

Zhu S, Cao R, Yu K (2020) Dual learning for semi-supervised natural language understanding. IEEE/ACM Trans Audio Speech Lang Process 28:1936–1947. https://doi.org/10.1109/TASLP.2020.3001684

Ziebart BD, Maas A, Bagnell JA, Dey AK (2008) Maximum entropy inverse reinforcement learning. In: 23rd National conference on artificial intelligence (AAAI), Chicago, IL, USA, vol 3. AAAI Press, pp 1433–1438