Application of Neurosymbolic AI to Sequential Decision Making – Dissertation Abstract

Carlos Núñez-Molina[1]
Juan Fernández-Olivares[2] (PhD director), Pablo Mesejo[3] (PhD co-director)
[1,2,3]Universidad de Granada, Spain
[1]ccaarlos@correo.ugr.es, [2]faro@decsai.ugr.es, [3]pmesejo@decsai.ugr.es

Abstract

In the history of AI, two main paradigms have been applied to solve Sequential Decision Making (SDM) problems: Automated Planning (AP) and Reinforcement Learning (RL). Among the many proposals to unify both fields, the one known as *neurosymbolic AI* has recently attracted great attention. It combines the Deep Neural Networks characteristic of modern RL with the symbolic representations typical of AP. The main goal of this PhD is to progress the state of the art in neurosymbolic AI for SDM. To do so, three different lines of research have been proposed. In the first one, I will perform a study of the literature and summarize my findings into a review. In the second one, I will extend my previous work (Núñez-Molina et al. 2021), which combined Deep Q-Learning with Classical Planning to improve planning performance, with the ability to manage non-determinism and will apply it to a real logistics problem. Finally, in the third line of research, I will develop a method for generating planning problems and leverage it to learn HTN domains without expert traces and to study the properties of planning domains.

Introduction

Sequential Decision Making (SDM) (Littman 1996) is the problem of solving Sequential Decision Processes (SDP). In a SDP, an agent situated in an environment must make a series of decisions in order to complete a task or achieve a goal. These ordered decisions must be selected according to some optimality criteria, generally formulated as the maximization of reward or the minimization of cost. SDPs provide a general framework which has been successfully applied to solve problems in fields as diverse as robotics (Kober, Bagnell, and Peters 2013), logistics (Schäpers et al. 2018), games (Mnih et al. 2015), finance (Charpentier, Elie, and Remlinger 2021) and natural language processing (Wang, Li, and He 2018).

Throughout the years, many AI methods have been proposed to solve SDPs. They can be grouped in two main categories: Automated Planning (AP) (Ghallab, Nau, and Traverso 2004) and Reinforcement Learning (RL) (Sutton and Barto 2018). These two paradigms mainly differ in how they obtain the solution and how they represent their knowledge:

- AP techniques exploit the existing prior knowledge about the environment dynamics, encoded in what is known as the action model or planning domain, to carry out a search process in order to find a valid plan that achieves a set of goals starting from the initial state. Regarding the knowledge representation employed, many AP techniques encode their knowledge in a symbolic manner, using a formal language based on first-order logic such as PDDL (Fox and Long 2003). These symbolic AP methods will be referred to as Symbolic Planning (SP).
- Standard RL methods learn the optimal policy, i.e., a mapping from states to actions in order to maximize reward, automatically from data, without performing any planning process whatsoever. Additionally, the vast majority of RL methods represent their learned knowledge in a subsymbolic way, often as the weights of a Deep Neural Network (DNN).

Each of these paradigms presents several advantages and drawbacks:

- In the case of SP, its symbolic representation is amenable to interpretation, whereas most RL methods represent their knowledge as a non-interpretable black-box. Additionally, SP methods can more easily reason about long-term sequences of actions than RL approaches, which tend to be greedier.
- The main advantage of RL is that it is able to learn the solution of the SDP, i.e., the optimal policy, from data automatically obtained by interacting with the environment. Due to this, RL can be easily applied to solve a wide range of problems with little effort from the human engineers. In the case of SP, these human engineers must provide extensive prior knowledge, mainly in the form of the planning domain, which requires a lot more effort.

Since the shortcomings of SP align with the strengths of RL and vice versa, it seems natural to try to reconcile these two competing paradigms into a single unified approach.

Many works have tried to bridge the gap between RL and SP. Some of these works have focused on extending the capabilities of RL methods with those of SP. Model-based RL (Moerland, Broekens, and Jonker 2020) enhances (model-free) RL with the ability to perform a look-ahead process over a model of the world. Additionally, several model-based RL methods try to learn how to actually carry out this deliberative process, i.e., they learn to plan (Tamar et al. 2016). Relational RL (Tadepalli, Givan, and Driessens 2004) represents the learned knowledge in a symbolic way, with the goal of improving the interpretability and generalization of classical RL. Regarding modern Deep RL, some works have substituted standard DNNs for architectures with relational inductive biases, such as Graph Neural Networks (GNN) (Battaglia et al. 2018), which represent the learned knowledge in terms of objects and their relations.

In parallel to all these efforts, many authors have taken inspiration from RL in order to improve SP. Some works have proposed efficient planning algorithms, such as MonteCarlo Tree Search (MCTS) (Kocsis and Szepesvári 2006), which do not require a symbolic representation of the environment dynamics. In the case of MCTS, the authors employ a RL algorithm known as UCB1 in conjunction with *rollouts* to estimate future rewards and guide the planning process. Other works have developed methods for learning the prior knowledge SP techniques require, in order to relieve some of the burden on experts. Some of them focus on learning the structure of the SDP, whose most important aspects are encoded in the planning domain (Segura-Muros, Pérez, and Fernández-Olivares 2021), whereas others learn domain-specific control knowledge to guide and speed up the search, such as planning policies and heuristics (Jiménez et al. 2012).

In recent years, a novel approach for integrating SP and RL has attracted a lot of attention. This hybrid approach, known with the name of *Neurosymbolic AI* (Besold et al. 2017), consists of methods which combine the DNNs usually employed in modern RL with the symbolic representations typical of SP. Neurosymbolic methods pose a promising approach towards a real unification of SP and RL, since they try to integrate the main strenghts of each approach: the ability to automatically extract knowledge from data as in RL with the interpretability and reasoning ability of SP.

Several works have successfully applied these methods to the field of SDM. (Asai and Fukunaga 2018) proposes *Latplan*, a model that uses a Variational Autoencoder (VAE) (Kingma and Welling 2013) to learn a symbolic action model from pairs of images, which makes possible to apply standard symbolic search algorithms to solve planning problems encoded in a subsymbolic way. (Katz et al. 2021) presents a semi-automatic method for performing scenario planning, i.e., generating a variety of possible future scenarios to support decision making and risk management. It uses DNNs to extract forces and causal relations from documents, and off-the-shelf planners that exploit the extracted information to generate plans corresponding to possible future scenarios. (Shen, Trevizan, and Thiébaux 2020) proposes STRIPS-HGNs, a special type of GNN that receives as input the symbolic description of a planning problem and predicts the heuristic value, which is then used to guide a symbolic planning procedure. Finally, (Toyer et al. 2018) presents ASNets, a novel type of DNN architecture which is able to learn a policy that generalizes to different problems of the same planning domain, thanks to its unique structure which resembles a symbolic planning process.

In the light of these recent advances, the main goal of this PhD dissertation is to progress the state of the art in neurosymbolic AI for SDM, with the development of methods for both solving SDPs and learning aspects of their structure.

Current State of the PhD

So far, the line of research has focused on the integration of SP and Deep RL. In (Núñez-Molina, Fdez-Olivares, and Pérez 2020), we proposed a neurosymbolic online execution system which combines PDDL-based planning with the Deep RL algorithm known as Deep Q-Learning (Mnih et al. 2013), in order to decrease the load of the planner. To perform this combination, we resorted to goal selection. We trained the Deep Q-Learning algorithm to select goals instead of actions, where the available goals are known a priori. Once a goal is selected, it is given to the symbolic planner, which obtains a plan from the current state to the goal. After achieving the goal, Deep Q-Learning is used to select a new one, and this process of interleaving planning and goal selection continues until the final goal is achieved. Instead of maximizing reward, Deep Q-Learning is trained to select goals in such a way that minimizes the length of the plans obtained by the planner. We tested our architecture on a deterministic version of the video game known as BoulderDash, present in the GVGAI framework (Perez-Liebana et al. 2015), where the agent must obtain a number of gems (which correspond to the goals in our method) and then exit the level. We trained our model on a set of training levels and evaluated it on a different set of test levels, in order to test its generalization ability. In this preliminary work, we compared the performance (in terms of plan length) of our model, referred to as *DQP*, with a baseline model trained with supervised learning to select the best next goal in a greedy way. The obtained results show that, as training data increases, our DQP model performs better than the greedy baseline model, since it performs long-term thinking when selecting goals. The results were presented in the ICAPS 2020 IntEx/GR workshop.

In our next work (Núñez-Molina et al. 2021), we greatly improved the performance of our approach regarding goal selection quality. In addition, we compared it against the same planner our architecture uses, but this time with no goal selection at all. The results obtained show that, on (geometric) average, our method is able to greatly reduce planning times in exchange for obtaining plans with 25% more actions. Thus, our approach strikes a good balance between time complexity and solution quality, and makes possible to apply standard symbolic planners to environments with tight time restrictions.

Then, in (Núñez-Molina, Fdez-Olivares, and Pérez submitted) we improved once again the performance of our approach, augmented our architecture with the ability to detect non-achievable goals, provided a more detailed mathematical formulation and compared our approach against standard Deep Q-Learning (with no planning). Results showed that our method generalizes better than standard Deep Q-Learning and is more sample-efficient. Thus, our neurosymbolic approach performs better than standard SP and RL when both plan quality and time requirements are considered. This third work has been submitted to a journal and is currently undergoing minor revisions.

Research Plan

The main research hypothesis explored in this PhD is the following: **Neurosymbolic AI provides a successful approach for solving and learning the structure of SDPs**. In order to verify this hypothesis, six goals corresponding to three different lines of research have been proposed.

The first line of research contains Goal 1 and simply corresponds to a study of the literature. The second one contains Goals 2 and 3 and continues the line of research on goal selection with Deep Q-Learning. The third one entails a method for automatically generating planning problems (Goal 4) and two possible applications of such method to learn aspects of the SDP structure (Goals 5 and 6). Although these three lines of research can be pursued in no particular order, the goals within each of them must be achieved following the goal numeration described below:

Goal 1: Study of the state of the art. Firstly, I will perform a comprehensive study of the state of the art regarding AI methods for SDM, focusing on the case of finite Markov Decision Processes (MDP) (Sutton and Barto 2018). I will research AI methods ranging from standard SP and RL/Deep RL techniques to hybrid approaches such as model-based RL, techniques for *learning to plan* and neurosymbolic models. I will not only study how these techniques can be applied to solve MDPs, i.e., finding the optimal policy/plan, but also how they make possible to learn the structure of the MDP, e.g., the action model and additional aspects such as landmarks (Hoffmann, Porteous, and Sebastia 2004).

This first goal has already been partially completed. As detailed above, I have researched the field of AI for SDM and summarized my findings into a review which I intend to submit very soon. In this review, I have given special emphasis to those hybrid models which try to integrate the competing SP and RL approaches. This work is also part position paper, as I discuss what properties an ideal method for SDM should possess. Based on these ideal features, I propose a theoretical framework to compare the different methods for solving SDPs. As a result of this comparison, I conclude that neurosymbolic models are currently the closest approach to this ideal method for SDM and, thus, pose a very promising direction for future work.

Goal 2: Uncertainty management with DQP. In this goal, I plan to continue my previous line of research on goal selection with Deep Q-Learning. My intention is to enhance my DQP model with the ability to manage uncertainty, so that it can be applied to stochastic environments. To achieve this, I will employ Deep Q-Learning to predict the uncertainty associated with a given goal in addition to the plan length. This uncertainty will correspond to the probability that a plan from the current state to the goal can be successfully executed from start to finish without interruptions (e.g., an obstacle suddenly appearing). This way, it will be possible to select those goals which result in a short plan overall in addition to being likely achievable by the agent, which results essential for dynamic, non-deterministic environments.

This new ability of the DQP model also serves as an execution monitoring tool. By repeatedly predicting the uncertainty associated with an already-selected goal during the execution of its plan, the agent will be able to detect unexpected situations as a result of an increase in the uncertainty value predicted by the network. Once this value surpasses a given threshold, we can consider the goal no longer achievable. In this case, the agent will simply select a new goal to replace the old one and try to achieve it.

One interesting property of this approach is that the goal selection procedure (based on Deep Q-Learning) is responsible for completely managing the uncertainty, i.e., non-determinism, in the environment. In principle, this makes possible to apply a classical, deterministic planner to a stochastic environment, instead of a probabilistic planner. However, it remains unclear if this approach (deterministic planner + goal selection with uncertainty) will be enough in highly dynamic environments or, instead, it will be necessary to also manage uncertainty at the planner level, i.e., by using a probabilistic planner.

Goal 3: Application of DQP to a real logistics problem. As the final step in this line of research, I plan to apply my DQP approach to solve a real-world problem. The goal is to manage the logistics of a company which relies on a fleet of trucks to transport and deliver packages. The main decisions to take concern which packages each truck must carry and the route each truck must follow in order to successfully deliver the packages to their recipients, while meeting a set of requirements such as time deadlines, service cost restrictions, and regulation constraints. Since this problem is too complex to be directly solved with standard SP techniques, I propose to use my DQP approach to map high-level decisions, such as to which truck assign each delivery order, to goals and train my goal selection method to select them so as to optimize a series of metrics, e.g., minimize the total distance traveled, and satisfy a set of criteria, e.g., each package must be delivered before its deadline. Once the goals have been selected, a symbolic planner can then be used to obtain the low-level plan that achieves each goal. Furthermore, the DQP model will be used to adapt to unexpected situations, such as new delivery orders and truck breakdowns. To achieve this, it will constantly monitor the state of the execution and modify the goals to achieve (and their associated plans) when needed.

Several modifications will need to be made in order to adapt the DQP model to this real-world scenario, many of which will only be clear once the specifications of the problem are well known. However, since the data for training the model will be in the form of logs, it seems likely that the DQP model will need to be able to handle symbolic data. Up to this point, the DNN trained with Deep Q-Learning to select goals corresponds to a Convolutional Neural Network (CNN) (Krizhevsky, Sutskever, and Hinton 2012), which receives an image-like encoding of the states and goals to select from. In order to adapt it to symbolic data, I plan to substitute the CNN with a DNN suitable for relational representations. A reasonable approach would be to employ a GNN and encode the information about the states and goals as a graph, which would then be given as input to the network.

Goal 4: Neurosymbolic method for generating planning problems. This goal corresponds to a new line of research, consisting on the automatic generation of planning problems for any given planning domain. This method will be useful for two main purposes: generating data for training Machine Learning methods (e.g., those that learn planning policies and heuristics) and creating a set of benchmark problems to compare the performance of different planners in planning competitions (such as those held in the ICAPS). I plan to develop a method which receives as input a PDDL domain and outputs a set of planning problems pertaining to that particular domain. The problems generated must satisfy three main properties: validity (the initial state must represent a possible situation of the world and the problem must be solvable), diversity (the problems must vary in kind) and quality (according to a metric defined by the user, such as the resolution difficulty of the problems).

In a similar fashion to (Fuentetaja and De la Rosa 2012), I will follow a *problem generation as planning* approach, where SP is used to generate planning problems. Each planning problem is composed of two parts: the initial state and the goal to achieve. For each problem, a valid initial state is first generated. Starting from an empty state, a search process is performed, where at each step either a new object is added or a predicate is instantiated on the state objects. After generating the initial state, a different search process is performed, where each step corresponds to applying a valid action of the planning domain and modifying the state predicates according to the action effects. Once a given number of actions have been executed, the problem goal is generated as a subset of the predicates of the state the search procedure has ended at. In addition to the planning domain, the user may provide two additional pieces of prior information: a validation method (e.g., a list of rules) which receives as input an initial state and returns whether it is valid or not (otherwise every possible initial state is regarded as valid), and the set of predicates which can form part of the goal (otherwise all the predicates will be considered).

The validation method is used to prune the search when a node is generated corresponding to a non-valid initial state. The other validity requirement, regarding the solvability of the generated problems, does not need to be checked, since the sequence of actions applied to obtain the goal from the initial state correspond to a valid solution plan. In order to generate problems of a certain quality, e.g., problems with properties which make them hard to solve, I will rely again on Deep Learning. A DNN (possibly a GNN) will receive as input a partially-generated problem (possibly encoded as a graph) and output its quality value, which will be used to guide the search towards problems of good quality. Once a problem has been completely generated, it will be solved with an off-the-shelf planner in order to assess its quality. For example, the resolution difficulty of a problem can be estimated as the number of states the planner needed to explore to find its solution. The computed metric will serve as a reward signal to train the DNN with RL methods. Finally, it is important that the generated problems are diverse. One possible approach is to obtain at each step the *n* nodes with highest quality and then select one of them completely at random. This provides a simple method for balancing quality and diversity which resembles the *epsilon-greedy* exploration method in RL.

Goal 5: Learning HTN domains from PDDL without plan traces. Here I plan to leverage the problem generation method previously explained in order to learn a Hierarchical Task Network (HTN) (Georgievski and Aiello 2015) domain just from a PDDL domain, with no need for plan traces. Given a planning domain, I will use the method in Goal 4 to generate a large and diverse set of planning problems. Then, the generated problems will be solved with an off-the-shelf planner and the solution plans obtained will be provided, along with the planning domain and problems, as inputs to some HTN learning method such as the one proposed in (Nejati, Langley, and Konik 2006). This way, it will be possible to learn a HTN domain just from a PDDL domain, without needing an expert to provide example problems and their solution plans, since these will be obtained with my problem generation method. If the generated problems properly represent the different types of problems in the domain, then the HTN domain obtained should be applicable to solve any of them. An alternative approach is to bias the problem generation method towards generating instances of a given type, e.g., problems of high difficulty. Then, the HTN domain learned from them should be tailored to this specific type of problems, thus helping to solve them in a very efficient way.

Goal 6: Domain characterization. As the final goal of my PhD, I plan to utilize the problem generation method previously detailed to study the properties of a particular planning domain, what I refer to as *domain characterization*. Given a planning domain, a large and diverse set of planning problems will be generated and then solved with an off-the-shelf planner. Once this data has been obtained, I will use Data Mining techniques to study the properties of the generated problems and their solutions, which will serve to characterize the domain as a whole. One possible approach is to apply clustering techniques to group the problems in clusters of similar problems according to a series of metrics, e.g., resolution difficulty, solution length and resources needed to solve the problem. By studying the properties of these clusters, it will be possible to analyze the different types of problems in the domain, which will help to understand the different situations which can arise in it. For example, in the logistics domain of Goal 3, each problem could correspond to a different logistics task, composed of a particular number of trucks, packages, locations and delivery orders. By clustering the problems according to their resource utilization (e.g., how many kilometers the trucks need to travel to deliver the packages), it will be possible to analyze which characteristics influence the utilization of resources (e.g., large packages and delivery orders for distant cities, etc.), which could be useful as a decision support system.

Acknowledgements

This work is being partially funded by the Andalusian Regional Projects B-TIC-668-UGR20 and PYC20-RE-049UGR, and the Spanish National Project RTI2018-098460-B-I00 with FEDER funds.

References

Asai, M.; and Fukunaga, A. 2018. Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Battaglia, P. W.; Hamrick, J. B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; et al. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*.

Besold, T. R.; Garcez, A. d.; Bader, S.; Bowman, H.; Domingos, P.; Hitzler, P.; Kühnberger, K.-U.; Lamb, L. C.; Lowd, D.; Lima, P. M. V.; et al. 2017. Neural-symbolic learning and reasoning: A survey and interpretation. *arXiv preprint arXiv:1711.03902*.

Charpentier, A.; Elie, R.; and Remlinger, C. 2021. Reinforcement learning in economics and finance. *Computational Economics*.

Fox, M.; and Long, D. 2003. PDDL2. 1: An extension to PDDL for expressing temporal planning domains. *Journal of artificial intelligence research*.

Fuentetaja, R.; and De la Rosa, T. 2012. A Planning-Based Approach for Generating Planning Problems. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Georgievski, I.; and Aiello, M. 2015. HTN planning: Overview, comparison, and beyond. *Artificial Intelligence*.

Ghallab, M.; Nau, D.; and Traverso, P. 2004. *Automated Planning: theory and practice*. Elsevier.

Hoffmann, J.; Porteous, J.; and Sebastia, L. 2004. Ordered landmarks in planning. *Journal of Artificial Intelligence Research*.

Jiménez, S.; De La Rosa, T.; Fernández, S.; Fernández, F.; and Borrajo, D. 2012. A review of machine learning for automated planning. *The Knowledge Engineering Review*.

Katz, M.; Srinivas, K.; Sohrabi, S.; Feblowitz, M.; Udrea, O.; and Hassanzadeh, O. 2021. Scenario planning in the wild: A neuro-symbolic approach. *FinPlan 2021*.

Kingma, D. P.; and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kober, J.; Bagnell, J. A.; and Peters, J. 2013. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*.

Kocsis, L.; and Szepesvári, C. 2006. Bandit based monte-carlo planning. In *European conference on machine learning*, 282–293.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*.

Littman, M. L. 1996. *Algorithms for sequential decision-making*. Brown University.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *nature*.

Moerland, T. M.; Broekens, J.; and Jonker, C. M. 2020. Model-based reinforcement learning: A survey. *arXiv preprint arXiv:2006.16712*.

Nejati, N.; Langley, P.; and Konik, T. 2006. Learning hierarchical task networks by observation. In *Proceedings of the 23rd international conference on Machine learning*.

Núñez-Molina, C.; Fdez-Olivares, J.; and Pérez, R. 2020. Improving online planning and execution by selecting goals with deep q-learning. In *ICAPS 2020 Workshop on Integrated Execution (IntEx) / Goal Reasoning (GR)*.

Núñez-Molina, C.; Fdez-Olivares, J.; and Pérez, R. submitted. Learning to select goals in Automated Planning with Deep Q-Learning. *Expert Systems With Applications*.

Núñez-Molina, C.; Vellido, I.; Nikolov-Vasilev, V.; Pérez, R.; and Fdez-Olivares, J. 2021. A Proposal to Integrate Deep Q-Learning with Automated Planning to Improve the Performance of a Planning-Based Agent. In *Conference of the Spanish Association for Artificial Intelligence*.

Perez-Liebana, D.; Samothrakis, S.; Togelius, J.; Schaul, T.; Lucas, S. M.; Couëtoux, A.; Lee, J.; Lim, C.-U.; and Thompson, T. 2015. The 2014 general video game playing competition. *IEEE Transactions on Computational Intelligence and AI in Games*.

Schäpers, B.; Niemueller, T.; Lakemeyer, G.; Gebser, M.; and Schaub, T. 2018. ASP-based time-bounded planning for logistics robots. In *Twenty-Eighth International Conference on Automated Planning and Scheduling*.

Segura-Muros, J. Á.; Pérez, R.; and Fernández-Olivares, J. 2021. Discovering relational and numerical expressions from plan traces for learning action models. *Applied Intelligence*.

Shen, W.; Trevizan, F.; and Thiébaux, S. 2020. Learning domain-independent planning heuristics with hypergraph networks. In *Proceedings of the International Conference on Automated Planning and Scheduling*.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Tadepalli, P.; Givan, R.; and Driessens, K. 2004. Relational reinforcement learning: An overview. In *Proceedings of the ICML-2004 workshop on relational reinforcement learning*.

Tamar, A.; Wu, Y.; Thomas, G.; Levine, S.; and Abbeel, P. 2016. Value iteration networks. *Advances in neural information processing systems*.

Toyer, S.; Trevizan, F.; Thiébaux, S.; and Xie, L. 2018. Action schema networks: Generalised policies with deep learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Wang, W. Y.; Li, J.; and He, X. 2018. Deep reinforcement learning for NLP. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*.