KI - Künstliche Intelligenz
https://doi.org/10.1007/s13218-024-00845-w

DISSERTATION AND HABILITATION ABSTRACTS

Comprehensible Extraction of Knowledge Bases for Learning Agents in Games

Daan Apeldoorn[1]

Received: 6 October 2023 / Accepted: 9 April 2024

Abstract
This dissertation abstract summarizes results of the thesis "Comprehensible Knowledge Base Extraction for Learning Agents - Practical Challenges and Applications in Games" (accepted as dissertation at the Department of Computer Science of TU Dortmund University, Germany). The thesis presents approaches that allow for the automated creation of knowledge bases from agent behavior learned in the context of games. The aims are twofold: (1) The creation of human-readable knowledge that can provide insights into what an agent learned, and (2) the investigation of how learning agents themselves can benefit from incorporating these approaches into their learning processes. Applications are presented, e.g., in the context of general video game playing. Moreover, an outlook on the InteKRator toolbox is provided which implements the most essential approaches in a more general context for the potential use in other domains (e.g. in medical informatics).

1 Introduction

More comprehensible artificial intelligence (AI) systems that are able to explain what they learn(ed) have gained more and more attention over the past years. In the dissertation *Comprehensible Knowledge Base Extraction for Learning Agents – Practical Challenges and Applications in Games* [1] (accepted at TU Dortmund University, Germany), this idea is considered in the context of learning agents with applications in games. The dissertation presents approaches that allow for the automated creation of knowledge bases from agent behavior. It aims especially at (1) the creation of human-readable knowledge for providing insights into what an agent has learned, and (2) the investigation of how learning agents can benefit from incorporating these approaches into their learning processes. Applications are presented, e.g., in the context of general video game playing (GVGAI). Moreover, approaches with potential for other domains have been implemented in the InteKRator toolbox.[1] The dissertation is mainly based on the publications [2–10]. A summary of contributions can be found in Table 1.

After a rough overview over related works (Sect. 2), this abstract briefly presents how knowledge bases can be extracted from learned agent behavior (Sect. 3) and how agents can benefit from that with respect to challenges in games (Sect. 4). As a practical result, the InteKRator toolbox is presented (Sect. 5) and a conclusion with an outlook on future work is provided (Sect. 6).

2 Related Work

Works related to the presented approaches, can be roughly categorized into the following groups (see [1] for details):

- learning approaches that are able to provide structural insights into what an agent learns
- approaches that are geared towards a *comprehensible* representation of knowledge (i.e., that is not only compact but also easy to read and understand for humans)
- learning/hybrid agent models in the context of games
- systems similar to the InteKRator toolbox.

Representatives of the first group are, e.g., Bayesian (or other probabilistic) networks (e.g., [14], Section 8.2.2) or decision trees. Although there are methods to learn the structure of a Bayesian network [15], the structure is oftentimes provided

✉ Daan Apeldoorn
daan.apeldoorn@uni-mainz.de

1 IMBEI Medical Informatics, University Medical Center of the Johannes Gutenberg University Mainz, Mainz, Germany

1 https://gitlab.com/dapel1/intekrator_toolbox.

Table 1 Summary of contributions, selected occurred difficulties and challenges as well as features and results

Contributions	Difficulties/challenges	Features/results
A multi-abstraction-level knowledge representation approach [5]	Evaluation of the comprehensibility	Efficient reasoning, human-readable representations (in the study [10], reasoning efficiency and comprehensibility were higher in comparison to ASP [11])
Approaches for learning such representations from data [3, 4]	Efficiency of learning; proving the completeness	Two algorithms: a preliminary [4] and a more efficient one [3], a completeness proof of the latter in [1]
A measure for the subjectively experienced strategic depth in games [8]	An eligible definition of the measure and its evaluation	A good fit to the strategic depth felt by humans players in [8] when playing GVGAI games
Two learning agent models [6, 9] incorporating some of the approaches	The GVGAI framework's strong time constraints (cf. footnote 4)	Learning of formal forward models in GVGAI games [2, 9] based on which MCTS (see, e.g., [12]) and similar methods have been applied successfully
Practical implementations to be used in further domains [7]	Keeping interfaces lightweight; handling of numeric data	The open-source toolbox InteKRator [13] combining knowledge representation and machine learning aspects

in advance and most graphical approaches can become hard to visualize and read for larger amounts of nodes.

As for the second group, the concept of *defaults* (as described by Reiter in his *Default Logic* [16]) provides interesting properties for covering larger amounts of knowledge, since many common cases can be covered with a single default, leaving the remaining cases to few more specific rules. *Answer Set Programming* (ASP; see, e.g., [11]) allows for implementing these kinds of ideas by offering two different kinds of negation (*strict* vs. *default*). The approach presented here exploits these ideas on multiple levels of abstraction. Thereby, it allows for creating compact knowledge bases that were easier to read and comprehend by humans in comparison to ASP in the joint study [10]. In the presented approach, the rules are quantified by weights, which serves as an interface to machine learning approaches when learning such representations from data.

In the third group, the discipline of *General Video Game Artificial Intelligence* (GVGAI) [17], where agents must learn to play different (a priori unknown) video games, represents a challenging application domain. The *GVGAI competition* represents benchmarks for agent models in this context. Besides their sensory inputs, agents may be provided with a forward model of a game (i.e., a model that allows for extrapolating future states) or must learn the game mechanics themselves—in addition to good playing/winning strategies. While in the first case, known methods such as *Monte Carlo Tree Search* (MCTS) [12] can be directly applied, the methods in the dissertation focus on the latter case.

Concerning the fourth group, other systems exist that cover either machine learning or knowledge representation approaches. One representative of each group will be briefly considered in the following (both being Java-based like the InteKRator toolbox): Weka [18] is a collection that covers a large amount of different machine learning packages. However, Weka is not explicitly geared towards knowledge representation techniques. In contrast, the collection provided by the TweetyProject [19] comprises a large amount of mainly logic-based approaches. However, machine learning aspects are not in the focus there and only one of the packages is explicitly related to machine learning. Unlike these established collections, the InteKRator toolbox focuses on the lightweight usage of knowledge representation concepts in combination with learning approaches.

3 From Agent Behavior to Knowledge Base

The principle idea here is to describe the behavior that an agent learned in its environment in the form of formal knowledge. The knowledge describing the learned behavior should be both compact and general in a way that keeps it readable for humans while at the same time, allowing for generalizations. In contrast to classical approaches, such as first-order logic, the idea of *default rules* [16] as well as the *default negation* in ASP [11] allow for covering many common cases of an agent's environment by focusing on the exceptional cases. Moreover, non-monotonicity is connected to games since games are highly dynamic environments: as an example, new levels may introduce large changes to the game play while an agent's overall default behavior (e.g., reaching a goal) might still be useful. The approach presented here is based on the idea of *rules with exceptions* (and exceptions of exceptions, etc.) where every exception to a rule is a rule itself with a more specific premise. Thereby, a knowledge hierarchy from general to more specific rules is induced that can be read top-down to gain insights into what an agent has learned.

Different algorithms are presented in [1] to learn such representations from data, i.e., from sequences (or "traces") of state-action pairs (st, a) that have been produced by an agent's learning process. Every st is considered a conjunction $s_1 \wedge \cdots \wedge s_n$ where every $s_i \in \mathbb{S}_i$ is a value of the agent's ith sensor.[2] The following example tries to provide an intuition of the main ideas (similar examples can be found, e.g., in [1] or [8]).

Example 1 An agent in a grid world with state space $\mathbb{S} := \mathbb{S}_x \times \mathbb{S}_y$ and action space $\mathbb{A} := \{\text{Left}, \text{Right}, \text{Up}, \text{Down}\}$ has learned to navigate around an obstacle, starting from a state $st_{\text{start}} := x_0 \wedge y_0$ to a destination state $st_{\text{dest}} := x_7 \wedge y_0$. The state-action sequence resulting from the agent's trace through the grid world is assumed to be[3]:

$$\begin{aligned} SA = \langle\ & (x_0 \wedge y_0, \text{Up}), \dots, (x_0 \wedge y_4, \text{Up}), \\ & (x_0 \wedge y_5, \text{Right}), \dots, (x_6 \wedge y_5, \text{Right}), \\ & (x_7 \wedge y_5, \text{Down}), \dots, (x_7 \wedge y_1, \text{Down})\ \rangle \end{aligned}$$

A knowledge base learned from SA can then look as follows:

$$\begin{aligned} KB = \langle\ & \{\top \rightarrow \text{Right } [0.41]\}, \\ & \quad \{x_0 \rightarrow \text{Up } [0.83], \\ & \quad\ x_7 \rightarrow \text{Down } [1.0]\}, \\ & \qquad\quad \{x_0 \wedge y_5 \rightarrow \text{Right } [1.0]\}\ \rangle \end{aligned}$$

with annotated weights $[w]$ representing conditional relative frequencies $w := P(\mathit{conclusion} \,|\, \mathit{premise})$.

The algorithm used here to learn KB adds at first the topmost rule $\top \rightarrow \text{Right } [0.41]$ (since Right is the most common action in SA). Afterwards, it tries to find exceptions to cover as many cases as possible from SA, resulting in the two rules for Up and Down on the second level. The second-order exception on the bottommost level is added last to cover the case of the grid world's upper left corner (state $x_0 \wedge y_5$) where the agent moved Right (instead of Up). (For a more general explanation of the algorithm, see Sect. 5.)

Starting from the most general level, KB can be read topdown as: "*Usually go to right, except when perceiving x_0 then go up, or when perceiving x_7 then go down, except when perceiving $x_0 \wedge y_5$ then go right.*"

4 Benefit for Learning Agents

Apart from being used for explaining agent behavior, it has been investigated how learning agents can benefit themselves from incorporating knowledge base extraction approaches into their learning process. The aforementioned approaches have been integrated into two different agent models:

- a reinforcement/Q-learning-based [20] model [4, 6]
- an agent model learning a formal *forward model* of its environment that describes for a provided state st and an action a the expected subsequent state st' [2, 9].

In the first model, reinforcement learning is integrated with the extraction of a knowledge base from agent behavior during the learning process (cf. Example 1). In the context of different grid world scenarios as well as in a GVGAI game, it was shown that agents benefited already in an early stage of the learning process ($\approx$ 10 to 15% of the process, according to [1]) from relying their decisions on the extracted knowledge base (rather than on the weights learned through the underlying reinforcement learning approach). Such agents showed an increased learning speed over pure Q-learning in the experiments of [4, 6].

The second model allows an agent for learning forward models, which are then used to apply techniques such as *Monte Carlo Tree Search* (MCTS) [12]. Since many games are (near) real-time environments, games often have special performance requirements.[4] Moreover, in the GVGAI competition, agents are usually trained on certain levels of a game and then evaluated on other levels of the same game. To overcome these challenges, the agent model combines forward model learning from observational data with a revision approach: while learning a forward model in the training phase, in the evaluation phase, the learned forward model is revised when observing new effects that do not conform to the learned model. In our experiments [2, 9], the agent model was able to quickly learn human-readable forward models of GVGAI games based on which, e.g., MCTS were successfully applied, allowing the agent for learning to play several GVGAI games from scratch (performance videos can be found in the dissertation's online appendix [21]).

[2] In case of forward model learning (see Sect. 4), it is learned from state-action conjunctions and subsequent states $(st \wedge a, st')$ instead.

[3] Note that, unlike in [1], ordered sets are denoted here by $\langle \dots \rangle$.

[4] In the GVGAI competition's learning track, according to the rules prior to 2018, the learning time was limited to 5 min and decision-making was limited to 40 ms.

5 The InteKRator Toolbox

To make the results of this work more accessible to a broader range of applications (also beyond the scope of agents), central concepts have been implemented in the InteKRator toolbox [13]. The toolbox allows for learning comprehensible knowledge bases from data, performing reasoning on these knowledge bases and revising the learned knowledge bases with new evidence. Moreover, InteKRator provides the possibility to check a learned (or manually modeled) knowledge base against a data set for measuring the quality of the knowledge. InteKRator has been proposed to be used in the medical domain [7] and has recently been used in research for cancer therapy recommendations [22]. The main functionalities of InteKRator will be briefly considered in the following:

Learning Algorithm Inputs are a data set of n input and 1 outcome column and the output is a learned knowledge base $KB = \langle R_1, \dots, R_{n+1} \rangle$ (cf. Example 1). It starts with the topmost level $R_1 \in KB$ by adding a rule with an empty premise whose conclusion reflects the majority of the values contained in the outcome column. After that, on the next level, rules of premise length 1 are added, representing exceptions to the rule on the topmost level. This is continued successively, such that rules on a level R_j represent exceptions to the rules on the levels $R_{j'<j}$ (i.e., levels above R_j).

Reasoning Inputs are a knowledge base KB and a set of (assumed) evident knowledge. The algorithm outputs the inference(s) that could be derived from KB with the help of the evident knowledge (optionally together with the rule(s) from which the inference(s) are derived). The reasoning algorithm searches the knowledge base upwards, starting on the bottommost level, for the most specific rule whose premise is satisfied by the provided evident knowledge and returns its conclusion. In case of multiple rules with the same weights are activated, all their conclusions are returned.

Revision Inputs are a knowledge base KB and the new knowledge (in the form of one or more input values and one outcome value). The output is the revised knowledge base KB'. In case the outcome value cannot be derived from KB with the provided input values, the algorithm removes the rule providing the wrong conclusion. If the outcome still cannot be derived, a new rules is added to KB based on the provided input and outcome values. Although the algorithm can in principle revise on any level $R_j \in KB$, only revision on the bottommost level R_{n+1} ensures that the new knowledge is incorporated without any side-effects.

Checking Inputs are a knowledge base KB and a data set, and the output are the percentages of data rows for which the outcome could be correctly derived from the input values.

By using the same knowledge base format and by providing a generic interface, these functionalities can be easily combined: learned knowledge bases can be revised and reasoning can be performed on the resulting knowledge bases.

6 Conclusion and Future Work

This dissertation abstract summarized how knowledge can be represented in a comprehensible way and how to learn such representations from (sensory) data. The major results of the dissertation comprise a comprehensible knowledge representation approach, a complete learning algorithm for learning knowledge bases from data as well as efficient revision and reasoning algorithms. These approaches have been used successfully for GVGAI research as well as for the implementation of the InteKRator toolbox to be used in further domains such as medical informatics. Future work could be, e. g., an investigation of when agents should revise extracted knowledge bases rather than relying on the underlying learning approach to quickly adapt to changes of an environment. Moreover, an extension of the inference approach, a study on its inference properties as well as the further development of the InteKRator toolbox could be interesting directions.

Acknowledgements I would like to thank my thesis supervisor Gabriele Kern-Isberner for supporting me over a long period of time. The abstract and future work in the context of InteKRator are partly funded by Carl Zeiss Foundation, Grant No. P2021-02-014 (TOPML).

Funding Open Access funding enabled and organized by Projekt DEAL.

Data availability The dissertation abstract onlyprovides a summary of the results based on the cited papers containing the original research. This research hasbeen conducted over a longer period of time and apart from what can be found in the orginal papers, it might bevery hard (or even impossible) to gather and refer to further data here.

References

1. Apeldoorn D (2023) Comprehensible Knowledge Base Extraction for Learning Agents – Practical Challengens and Applications in Games, Dissertation at TU Dortmund University. Mainz (publisher), Aachen. https://doi.org/10.25358/openscience-9303
2. Apeldoorn D, Dockhorn A (2021) Exception-tolerant hierarchical knowledge bases for forward model learning. IEEE Trans Games 13(3):249–262
3. Apeldoorn D, Hadidi L, Panholzer T (2021) Learning behavioral rules from multi-agent simulations for optimizing hospital processes. In: Chomphuwiset P, Kim J, Pawara P (eds) Multi-disciplinary Trends in Artificial Intelligence - 14th International Conference, MIWAI 2021, Virtual Event, July 2–3, 2021, Proceedings. Springer, Cham, pp 14–26
4. Apeldoorn D, Kern-Isberner G (2016) When should learning agents switch to explicit knowledge? In: GCAI 2016. 2nd Global Conference on Artificial Intelligence, EPiC Series in Computing, vol. 41, pp. 174–186. EasyChair Publications
5. Apeldoorn D, Kern-Isberner G (2017) Towards an understanding of what is learned: Extracting multi-abstraction-level knowledge from learning agents. In: V. Rus, Z. Markov (eds.) Proceedings of the Thirtieth International Florida Artificial Intelligence Research Society Conference, pp. 764–767. AAAI Press, Palo Alto, California
6. Apeldoorn D, Kern-Isberner G (2018) An agent-based learning approach for finding and exploiting heuristics in unknown environments. In: A.S. Gordon, R. Miller, G. Turán (eds.) Proceedings of the Thirteenth International Symposium on Commonsense Reasoning, London, UK, November 6-8, 2017, *CEUR Workshop Proceedings*, vol. 2052. CEUR-WS.org, Aachen
7. Apeldoorn D, Panholzer T (2021) Automated creation of expert systems with the intekrator toolbox. Stud Health Technol Inform 283:46–55
8. Apeldoorn D, Volz V (2017) Measuring strategic depth in games using hierarchical knowledge bases. In: 2017 IEEE Conference on Computational Intelligence and Games (CIG), pp. 9–16. IEEE, New York
9. Dockhorn A, Apeldoorn D (2018) Forward model approximation for general video game learning. In: C. Browne, M.H.M. Winands, J. Liu, M. Preuss (eds.) Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games (CIG'18), pp. 425–432. IEEE, Piscataway
10. Krüger C, Apeldoorn D, Kern-Isberner G (2017) Comparing answer set programming and hierarchical knowledge bases regarding comprehensibility and reasoning efficiency in the context of agents. In: Proceedings of the 30th International Workshop on Qualitative Reasoning (QR 2017) at International Joint Conference on Artificial Intelligence (IJCAI 2017) in Melbourne, Australia. Northwestern University, Evanston, Illinois
11. Brewka G, Eiter T (2011) Truszczyński: Answer set programming at a glance. Commun ACM 54(12):92–103
12. Browne CB, Powley E, Whitehouse D, Lucas SM, Cowling PI, Rohlfshagen P, Tavener S, Perez D, Samothrakis S, Colton S (2012) A survey of Monte Carlo tree search methods. IEEE Trans Comput Intell AI Games 4(1):1–43
13. InteKRator Toolbox (visited on Oct 5th, 2023). https://gitlab.com/dapel1/intekrator_toolbox
14. Borgelt C, Braune C, Kruse R (2020) Unsicheres, impräzises und unscharfes Wissen. Handbuch der Künstlichen intelligenz. De Gruyter Oldenbourg, Berlin, pp 279–341
15. Fierens D (2008) Learning Directed Probabilistic Logical Models from Relational Data (Het leren van gerichte probabilistisch-logische modellen uit relationele gegevens). Dissertation. Lirias, Katholieke Universiteit Leuven, Leuven
16. Reiter R (1980) A logic for default reasoning. Artif Intell 13(1–2):81–132
17. Perez-Liebana D, Lucas SM, Gaina RD, Togelius J, Khalifa A, Liu J (2019) General video game artificial intelligence. Morgan and Claypool Publishers, San Rafael
18. Frank E, Hall MA, Witten IH (2016) The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques". Morgan Kaufmann, Fourth Edition
19. TweetyProject (visited on Mar 4th, 2024). https://tweetyproject.org
20. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction, 2nd edn. MIT Press, Cambridge
21. Online Appendix to Knowledge Base Extraction for Learning Agents (visited on Mar 6th, 2024). https://gitlab.com/kb-extraction-for-learning-agents/online-appendix
22. Thevapalan A, Apeldoorn D, Kern-Isberner G, Meyer RG, Nietzke M, Panholzer T (2023) Comparison and incorporation of reasoning and learning approaches for cancer therapy research. Stud Health Technol Inform 307:161–171