Association Rule Mining for Unknown Video Games

Alexander Dockhorn, Chris Saxton, and Rudolf Kruse

Otto-von-Guericke University,
Institute for Intelligent Cooperating Systems,
Universitaetsplatz 2, 39106 Magdeburg, Germany
{alexander.dockhorn, chris.saxton, rudolf.kruse}@ovgu.de

Abstract. Computational intelligence agents can reach expert levels in many known games, such as Chess, Go, and Morris. Those systems incorporate powerful machine learning algorithms, which are able to learn from, e.g., observations, play-traces, or by reinforcement learning. While many black box systems, such as deep neural networks, are able to achieve high performance in a wide range of applications, they generally lack interpretability. Additionally, previous systems often focused on a single or a small set of games, which makes it a cumbersome task to rebuild and retrain the agent for each possible application. This paper proposes a method, which extracts an interpretable set of game rules for previously unknown games. Frequent pattern mining is used to find common observation patterns in the game environment. Finally, game rules as well as winning-/losing-conditions are extracted via association rule analysis. Our evaluation shows that a wide range of game rules can be successfully extracted from previously unknown games. We further highlight how the application of fuzzy methods can advance our efforts in generating explainable artifical intelligence (AI) agents.

Keywords: planning, frequent pattern mining, association rule analysis, action selection, subgoal induction

1 Introduction

Artificial intelligence (AI) in games is known to have beaten expert human players in a wide range of boardgames. A common example is the chess playing agent Deep Blue [4], which defeated the at the time reigning world champion Garry Kasparov. The system was able to rate millions of board positions per second and was tuned in cooperation with multiple chess grandmasters. Recently, GO was added to this list with AlphaGO's victory over the world's best human GO player Lee Sedol [28]. Despite the great success of AlphaGO its implications are unclear. The utilized black-box model, a neural network based board evaluation function trained by reinforcement learning [8], is hard to interpret and highly specialized on a single task.

Despite the strength in playing these games, humans find it nearly impossible to interpret the result of black-box systems such as deep neural networks [27].

Here, the knowledge is represented in high-dimensional spaces learned from tons of data. The decision-making process cannot be explained easily and be fooled by adversarial examples. See, for example, the deep convolutional neural network architecture AlexNet[11]. Researchers were able to classify one of the largest image libraries “ImageNet Large Scale Visual Recognition Challenge” (LSVRC-2010; 1.2 million images, 1000 object classes) [25] with an astonishing top-5 error rate of 17.0%. However, recent analysis on the best performing solutions showed that minor perturbations can drastically change the classification result [30].

In the context of game AI similar problems can lead to unexpected behavior of the developed agent. For example, the system might fail in unusual situations or perform unreliably. During the interaction with such systems, players or users can have difficulties in explaining the AI’s behavior [5]. This can reduce the player’s immersion and, therefore, decrease the player’s enjoyment [29]. For this reason, the industry is in need of explainable AI systems, which act on the basis of comprehensible mechanisms. Crafting those systems by hand is an extremely costly and time-consuming task. Our work aims in developing such explainable AI systems by automatically adapting to the game environment.

While many proposed AI systems focus on generating an agent to play a single game, our work focuses on the task of creating an agent that is capable of learning to play multiple games. Thus, special attention goes to the generation of abstract learning systems, which are able to adapt to a wide range of game environments. Games provide a natural test bed for the task of general AI, because they exist in many variations and focus on providing a competing and balanced environment. Developing agents for games and especially for video games is of high interest due to the rapid growth of the (digital) entertainment industry. Learning AI agents would not just be able to play the game, but can adapt themselves to updates of the game throughout the development phase. Additionally, learning agents can be used for automatic game testing and balancing. Therefore, those systems would be able to drastically reduce time and costs of the development process.

The General Video Game AI (GVGAI) competition [21] provides games in a common description language, which eases the process of agent development and provides a unified benchmark for general purpose game AI. Previous results suggest that developed agents can achieve very good performance when they are provided with a forward model, which can be used to simulate the outcome of an agent’s action. However, this forward model needs to be hand-crafted and may not be available during the development of a game. For this reason, the in 2017 introduced learning track added a new challenge, in which agents only have a few tries for training and understanding the game without receiving any information on the game’s underlying rules.

In the learning track agents are provided with three levels during training, which introduce elements of the game being studied. The agent’s performance is subsequently validated based on its performance in two unknown levels. Most recent results of the 2017’s learning track competition show that developed agents performed barely better than an agent acting randomly. Proposed models include

Monte Carlo Tree Search (MCTS) agents with various parameter estimation algorithms [10]. This might be due to a lack of an understanding of the game's general rules and conditions for winning or losing the game.

In this work our goal is to improve the generation of a suitable game model during the training phase. Our system especially aims for an interpretable approximation of the game's rules and its winning or losing conditions, also known as termination set. This is achieved using play-trace analysis based on frequent pattern mining and association rule analysis. The complexity of the approximated rule set is reduced using multiple rating criteria. Our work introduces new ways of developing and validating agents for previously unknown games and is especially interesting for the development of interpretable models of the agent's environment. We further discuss possible fuzzy extensions and their influence on the interpretability of generated models.

The remainder of this paper will be structured as follows: In Section 2 we provide a basic overview of the topic general game AI and the framework we used to simulate a wide range of games. Further in Section 3, frequent pattern and association rule mining methods are reviewed, since they are the key elements in our model induction process. After explaining the basic concepts, we introduce our play-trace generating mechanism in Section 4. On the basis of these play-traces, we first use association rule analysis to model each game. Special constraints of the termination set are discussed in Section 5. Furthermore, we propose various rule reduction mechanisms. These are necessary to filter a set of interesting rules, which describe the game reasonably well. In Section 6 we present how explainable agents can be learned to highlight the strengths of the induction process. Subsequently, we discuss fuzzy-based extensions for the proposed framework in Section 7. Final remarks and interesting implications for future work can be found in Section 8.

2 General Video Game AI

General Game AI is a popular area of research. In contrast to previous works in game AI an agent is evaluated on its capability in playing a diverse set of games reasonably well. While humans can pick up simple games and learn their rules in a matter of minutes, most programmed agents need a lot of data to play the game on a competitive level. The General Video Game AI (GVGAI) competition [21] promotes further research on general games AI. It provides a framework for the creation of simple games with a similar back-end. Thus, agents can be tested on a wide range of video games without additional coding.

The GVGAI-framework is based on the works of Tom Schaul, who adapted the game definition language to the context of video games. His Video Game Definition Language (VGDL) [26] allows to quickly design games by providing entities and interactions of the game in the game description, while providing the actual level in a separate level description. The game description consists of: an interaction set, a termination set, a sprite set, and a level mapping [21].

The interaction set provides the basic model of the game. It describes how single sprites behave, how the player input influences the games avatar, and how interactions between those sprites change the current state of the game. During a collision it is possible to change the players score, to move, destroy or add sprites to the current game state, or change inherent attributes of represented game-objects, such as health points, movement speed or resources [24]. The interaction set also provides information about the physics engine used by each sprite. Currently two modes of operation are available: either grid-based physics, or continuous physics [22]. Grid-based physics simulates a top-down view in which sprites move from one grid cell to another or pixel-wise. Continuous physics were recently added to analyze learning models under more complicated physic environments. Here, sprites can move continuously in between grid-cells and simulate basic laws of physics such as gravity.

In this work we focus on modeling the agent's control options, the interaction set, as well as the termination set. The termination set defines a set of rules in which the game is destined to end. Possible mechanics include:

1. the count of a sprite is less/greater than a specified threshold, e.g., the player sprite was removed
2. a variable exceeds a specified threshold
3. an interaction occurs (which most often deletes a sprite and triggers the first option)

The GVGAI-Framework provides two observation mechanisms. Either the pixel output of the game engine is used, or the observation set is provided as an array-like representation of the game environment. In our work we will focus on the latter, which removes the influence of the sprite set.

The level mapping defines relationships between characters used in the level definition and sprites to be used. The level definition contains the start configuration of the game and is used to easily change the level structure without changing the rules of the game.

The GVGAI-Framework currently provides a set of 90 grid-based and 10 continuous-physics games. Researchers can participate in various competition tracks. In the playing track agents can access full information on the game model and the current game state. The learning track limits the agents to a partial observable game-state and does not provide a game-model. Our work focuses on the learning-track, in which the agent cannot use a forward-model. Therefore, our methods aim in generating a forward model based on observations during multiple play-traces.

3 Frequent Pattern Mining (FPM) and Association Rule Analysis

In contrast to black-box models we aim for an interpretable system. The output should be in human-readable format. Especially interesting are well compressed systems such as decision trees [23] and association rules. The strength of these methods is their output in form of human-readable rules. Those are not just easy to interpret but also easy to adapt. Due to the complexity of the game environments, we choose to implement an association rules analysis. In contrast to decision trees, association rule analysis works efficiently with a high number of features [3, 18]. Learning a tree would have a large overhead during the feature selection phase and result in a tree with high width and depth [9]. Additionally to the simplicity of these rule based systems, both can be adapted to non-deterministic (game) environments by using fuzzy systems. Both, fuzzy decision trees as well as fuzzy rule sets, would be possible extensions to our proposed solution. The utility of fuzzy extensions for the proposed approach is discussed in Section 7.

Our work relies on frequent pattern mining and association rule analysis [1], which results in simple rules in the form of:

$$\textit{antecedence} \rightarrow \textit{consequence}$$

Specifically, we are interested in rules that map the current game-state and possibly the agents action to the next game-state. This allows the agent to anticipate enemy movements, tile interactions and avoid any of the losing conditions of the game.

We achieve this by extracting patterns from recorded play-traces. A play-trace is an ordered set of observations of every tick of the game. Two consecutive game-ticks $(t_k,\ t_{k+1})$ will be merged to a single transaction, in which the action and each change in an observable element is included as an item $i \in B$ in the transaction. For each set of items I we count the support. The support of an item set is the number of transactions $t \in T$ that include the specified item set.

$$\text{supp}_T(I) = \big|\{t \in T \mid I \subseteq t\}\big| \cdot |T|^{-1}$$

In Frequent Pattern Mining (also called Frequent Item Set Mining) we search for all item sets with a support higher than a specified threshold, the minimum support supp_{min}. Therefore, the set of frequent items F is given by:

$$F(\text{supp}_{min}) = \{I \subseteq B \mid \text{supp}_T(I) \geq \text{supp}_{min}\}$$

After all frequent item sets were extracted, we create rules of the form $X \rightarrow Y$ for each item set in the set of frequent item sets ($I \in F$), such that $X \cup Y = I$ and $X \cap Y = \emptyset$. The set of rules will be filtered based on support and confidence of a rule.

The support of an association rule indicates how often a rule can successfully be applied in the given transaction database.

$$\text{supp}(X \Rightarrow Y) = \text{supp}(X \cup Y) = |\{t \in T \mid X \cup Y \in t\}| \cdot |T|^{-1}$$

It can be interpreted as the relative frequency in which the rule could have been applied. The confidence of an association rule describes how often the rule can be correctly applied in the given transaction database.

$$\operatorname{conf}(X \Rightarrow Y) = \frac{\operatorname{supp}(X \cup Y)}{\operatorname{supp}(X)}$$

It can also be interpreted as a conditional probability $P(E_Y|E_X)$, the probability of finding the consequence of the rule in a transaction in which the antecedence is known to be present.

4 Extracting Play-traces Patterns and Game Rules

Our algorithm starts with zero knowledge and tries to approximate the games rules by analyzing the play-traces of previous runs. Extracting play-traces can be a cumbersome task. In general those play-traces should cover all possible interactions that can occur during the game. However, without any knowledge you cannot perform much better than acting randomly. This can be seen in the results of 2017's GVGAI-learning track in which only one agent performed slightly better than a random agent. For this reason we generate an initial set of play-traces using a random agent. During play-trace collection the random agent is not learning from any previous experience. Therefore, all transactions in the database will be based on the same knowledge of the game, more precisely, in this special case on the absence of knowledge. Further play-traces can be collected using the generated model for enhancing the planning capabilities of an agent.

The capabilities of our method will be largely restricted by the set of features we store during our play-trace generation. Table 1 lists the recorded features used in this study. This list was iteratively adjusted to the needs of our method.

We use Frequent Pattern Mining to find frequently occurring patterns between consecutive time-steps. Especially important is the game result, which can be either *"continuing", "won",* or *"lost"*. In a play-trace that consists of n time-steps each time-step from $k = 1, \dots, n-1$ will be of the state "continuing" and only the last time-step is either marked with "won" or "lost". During longer games the support of only the ending time-steps will not reach a reasonable minimal support threshold. Therefore, no termination rules can be found in this case. Further, decreasing the minimal support to a level in which all termination rules can be found will result in far too many rules. However, we can make further use of special properties of the termination set for effective mining of such rules. These will be discussed in Section 5.

All other rules will be filtered based on support and confidence. While the support is used to reduce the number of frequent item sets, the confidence limits the rule base to rules which have a high probability of being correct. This is especially important because mistakes in our model will accumulate over time and even further reduce the accuracy of our prediction.

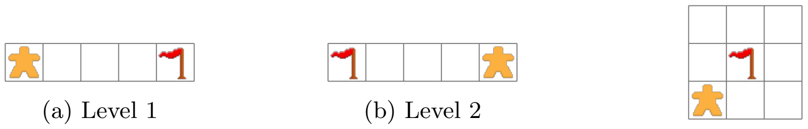
(a) Level 1 (b) Level 2 (c) Level 3

Fig. 1: Exemplary game where the player wins, when reaching the flag from the side and loses when he approaches from the top or the bottom.

Table 1: Recorded attributes during the play-trace analysis

Feature	Range
Game Tick	$\mathbb{N}$
Game Score	$\mathbb{N}$
Game Score Change	$\mathbb{N}$
Game Result	{Continuing, Won, Lost}
Player Action	{Up, Down, Left, Right, Use, other}
Player Grid-Position	$\{\mathbb{R}, \mathbb{R}\}$
Position Change X	$\mathbb{R}$
Position Change Y	$\mathbb{R}$
No Position Change	*boolean*
Player Collision	Object Type
Changes in Secondary Attributes such as Health, Resources, etc.	
Object Above	Object Type
Object Below	Object Type
Object Left	Object Type
Object Right	Object Type
Object Collision	{Sprite 1, Sprite 2}

5 Properties of the Termination Set

As a first example we consider the game depicted in Figure 1. The agent (yellow character) can move to neighboring grid-cells in each of the cardinal directions. He wins the game when he is approaching the flag from the side and loses when coming from the top or the bottom. In this game all winning play-traces in level 1 (Figure 1a) reach this point by moving to the right during their last step, while all play-traces in level 2 (Figure 1b) reach the target by moving to the left. By just looking at the final time-step-transition in all play-traces from the first and the second level five simple hypotheses are possible:

1. Moving right wins the game.
2. Moving left wins the game.
3. Reaching the flag wins the game.
4. Moving right and reaching the flag wins the game.
5. Moving left and reaching the flag wins the game.

Which of these is the most suitable hypothesis? Judging based on a single play-trace all seem equally valid. However, the hypothesis (1) and (2) can be excluded, in case we combine the observations of level 1 and level 2. Both levels will result in play-traces, which consist of multiple time-step-transitions in which a step to the right or to the left does not end the game. Deciding which of the other three hypotheses (3-5) is the most suitable cannot be done by just observing play-traces of a single level. Judging by complexity hypothesis (3) is the simplest and is correct for both levels. Both other hypotheses, (4) and (5), are valid for only one of the provided example levels. Therefore, we will return hypothesis (3) as a temporary result.

Nevertheless, in both levels there is only one way of fulfilling the general winning condition. In level 3, depicted in Figure 1c, there are multiple options for winning and losing the game. The player will instantly lose in case he approaches the flag from the top or the bottom.

The following hypothesis can be generated from analyzing single play-traces of the third level:

1. Reaching the flag wins the game.
2. Reaching the flag loses the game
3. Moving right and reaching the flag wins the game.
4. Moving left and reaching the flag wins the game.
5. Moving up and reaching the flag loses the game.
6. Moving down and reaching the flag loses the game.

All hypotheses for "Moving in any direction to win or lose the game." were removed, since, by the logic presented above, some play-traces will contain moves in the specified direction that did not end the game. From the generated hypotheses, (1) and (2) can be removed, because both contradict each other. None of the remaining generated hypotheses does fully describe the actual termination set. Nevertheless, we can combine hypotheses (3-6) in a termination set to fully describe the game's original model.

Using this method, only the last time-step will be used to form hypotheses for termination conditions. These hypotheses will be filtered by eliminating contradictions using the transactions of all previous time-steps. For the need of simplicity more complex hypotheses are exchanged by more simple termination conditions. Finally, the termination set can consist of multiple simple termination criteria. Further work needs to be put into the automatic compression of the termination set. This will ensure a faster processing during run-time and an improved readability in more complex termination sets.

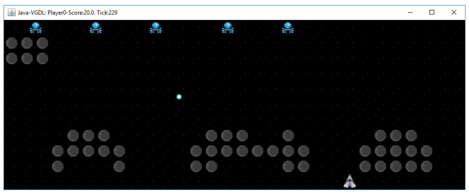

Fig. 2: Screenshot of the game alien of the GVGAI framework

6 Applications of Association Rule Analysis for General Game AI

The strengths of the system will be further highlighted based on multiple games of the GVGAI-framework. In this paper we present a detailed analysis of the game “alien”. For further results we refer to our current papers on the topic of forward model approximation [6, 7].

The game “alien” is a clone of the famous game space invaders. Here, the player controls a small spaceship and needs to defend itself from approaching aliens. Figure 2 shows a screenshot of the game. In the bottom row the player flies left or right. The spaceship can shoot a single bullet to destroy the alien spaceships before they arrive at the bottom of the screen. From time to time alien spaceships shoot as well. Their rockets and the player’s bullets explode at the gray rocks in the centre of the game board. The player wins by destroying all aliens, but loses by either destroying all rocks, or in case a single alien reaches the bottom row.

Play-traces were collected by letting the random agent play 10 rounds of “alien”. According to our proposed play-trace analysis we retrieved 8 rules. See Table 2 for the full list of rules.

Most of these rules are movement based, because movement occurs comparatively often. The lowered confidence of MoveRight appeared by coincidence, since the player always lost the game by moving right during the last time-step. During reset the game engine places the agent in the bottom left corner, therefore, a MoveLeft was detected. Interactions between sprites only occur rarely. For this reason, only one interaction rule can be found in the final rule set. The termination condition was successfully retrieved. However, since the agent did not win a game during 10 tries no rule for winning could have been retrieved. The final rule set describes the game reasonably well.

Table 2: Association rules extracted from play-trace analysis.

Rule			Support	Confidence
ActionLeft	$\rightarrow$	MoveLeft	0.32	1.00
ActionRight	$\rightarrow$	MoveRight	0.31	0.98
ActionUse	$\rightarrow$	Stay	0.36	1.00
ObjectAbove, ActionUse	$\rightarrow$	Stay	0.05	1.00
ObjectAbove, ActionLeft	$\rightarrow$	MoveLeft	0.05	1.00
ObjectAbove, ActionRight	$\rightarrow$	MoveRight	0.06	0.92
SpriteCollision	$\leftrightarrow$	HigherScore	0.06	1.00
PlayerCollision, ScoreDecrease	$\rightarrow$	GameLost	1.00	1.00

It is astonishing how well the systems grasps the true game model. Even without any further knowledge of the game, most necessary rules were detected. The resulting rule set is in human readable format and easy to understand. Agents can apply these rules in simulation-based search algorithms as approximation of the forward model. Chosen actions can later be justified by showing the user the related rules. Even the outcome of long action sequences can be explained by studying the agent's assumed outcome.

The result of this case-study also applies to many of the other games in the GVGAI-framework. A natural extension of this work, the Forward Model Approximation framework, can be found in [6, 7]. We also want to note that the development of game AI is just one of many interesting applications of this work. In general this work shows how explainable systems can be developed for unknown contexts by observation without providing further knowledge.

7 Analysis of Fuzzy methods

In this section we will discuss opportunities for the application of fuzzy methods to improve the proposed approach. Fuzzy logic is a many-valued extension of binary logic, which can be exploited in cases that allow truth values between true and false. This could be a desirable characteristic for multiple components of the designed model.

First of all the agent described in the current framework is described to be perfect in its observation of the environment. While the computer game setting often allows to implement agents with perfect sensors, transferring this approach to the field of robotics would need adjustments to the sensor value processing. Especially the item set mining needs to be fault-tolerant in cases where sensor values can be missing or does only receive vague information. The mining of gradual patterns can be used to act in scenarios with missing sources of information [16]. Also using fault-tolerant frequent item set mining algorithms [20], such as SODIM [2], will allow the application of the proposed approach in an error-prone setting. Such a fuzzy item set matching cannot only be applied during the mining phase, but also during the association rule processing phase. This may allow matching incomplete patterns for acting in situations in which none of the crisp rules apply.

Next to a fuzzy item set mining, the association rule mining process needs to be adapted for situations in which multiple rules may apply. This is especially relevant in non-deterministic games, in which a single situation can yield differing results. The proposed approach will generate conflicting association rules, which need to be handled by the agent during the decision-making process. By making the rule-exploitation fuzzy we allow the agent to weigh the applicability of multiple rules and their outcomes. Aggregating the results of these rules will allow the agent to act in situations in which the outcome of a situation can only be described by multiple degrees of possibility.

While the extensions above describe how the agent will be able to act in vague situations, fuzzy methods can also be applied to reduce complex rule sets to a human understandable form. Describing the final agent using a small set of fuzzy rules may help to understand the reasoning process and its resulting behavior. For example, fuzzy summaries can be used to reduce the fuzzy rule set to a small set of linguistic terms [13]. Therefore, applying fuzzy concepts cannot just help to adapt the proposed method to vague scenarios, but can also help to increase the explainability of the overall system behavior.

A final aspect, which is often ignored, is the interaction between the developed agent and a human user. We already mentioned the industry's interest in the development of self learning systems, but after the agent is adapted to the developed system it will, ultimately, interact with the human user. Measuring the user's emotions will inherently lead to subjective information. Managing and interpreting vague data gained from user questionnaires can be made possible by applying fuzzy data mining [15]. Sentiment analysis and opinion mining [19] will be the major research areas for increasing the user's enjoyment.

8 Conclusions and Outlook

In this work we discussed a rule induction mechanism for modeling unknown video games. Our case-study shows that simple and accurate rules can be extracted from a small set of observations. The learned model results in an approximation of the forward model. Utilizing this model helps to explain the agent's behavior and enables it to provide explanations of its intentions.

Our work highlights the capabilities of explainable machine learning systems, which can be used in complex environments without further knowledge. In contrast to other popular machine learning algorithms our proposed method results in a rule set readable by humans. This offers users the chance to further analyze the strengths and weaknesses of the developed agent and fosters the possibility of a strong human-computer interaction.

With this in mind many new research opportunities arise. For example, the possibilities for a stronger human-computer interaction would motivate tutoring systems, in which a machine teaches a new user based on the true model of an application and the model that can be induced from the user's behavior.

Especially interesting would be the generation of new learning examples, which help the user to get a grasp of the true model.

In contrast to teaching humans, the resulting model can also be used for a step wise increase in the agent's skill level. For this purpose, an agent would receive the induced rule set as an incomplete forward model. This would allow the agent to plan multiple steps ahead. Referring to our example game "alien", the agent would try to avoid collisions with other game entities to not lose the game. In combination with the extracted movement rules, the agent will be able to dodge bullets, thus, increasing its chances of winning the game. Collecting more play-traces with a partially informed agent will lead to an even better approximation of the true game model. Even without the true forward model, prediction of future game states could succeed with a reasonable enough approximation of the original model. This iterating approach will allow a step wise update of the agent's capabilities, ultimately allowing agents to learn how to be successful in a wide range of more complex applications, without any human interference.

Nevertheless, more complex models would also be in need of a more complex representation. Hierarchical rule representations and the mining of gradual patterns [16] would prove useful in preserving simplicity of the rule set in such scenarios. Another idea would be the development of local environment models. Especially multi-agent models might prove useful in modeling independent components of the environment. Such a distributed approach assures faster computation and higher memory efficiency. This could be achieved by applying the proposed approach to local observations of each entity of an environment. Selecting a scope of each local model and finding its relevant features will be a major focus due to the complexity of the environment.

Last but not least we see a special importance in the development of explainable models for non-deterministic applications. Those pose special constraints on the learning system, in which, for example, rules can trigger unreliably or have overlapping antecedences. Fuzzy models such as fuzzy decision trees [17, 14] and fuzzy rule sets [12, 16] may be the key to modeling such environments.

This selection of research opportunities does not aim to be complete, but it highlights the many directions in which explainable AI may advance during the next years. We are looking forward to all the upcoming applications, methods, and results that this field is about to offer.

References

1. Borgelt, C.: Frequent item set mining. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 2(6), 437–456 (2012)
2. Borgelt, C., Kötter, T.: Mining fault-tolerant item sets using subset size occurrence distributions. In: Gama, J., Bradley, E., Hollmén, J. (eds.) Advances in Intelligent Data Analysis X. pp. 43–54. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
3. Borgelt, C., Yang, X., Nogales-Cadenas, R., Carmona-Saez, P., Pascual-Montano, A.: Finding closed frequent item sets by intersecting transactions. In: Proceedings of the 14th International Conference on Extending Database Technology - EDBT/ICDT '11. p. 367. ACM Press, New York, New York, USA (2011)

4. Campbell, M., Hoane Jr., a.J., Hsu, F.h.: Deep Blue. Artificial Intelligence 134(1-2), 57–83 (2002)
5. Core, M., Lane, H., Van Lent, M., Gomboc, D., Solomon, S., Rosenberg, M.: Building explainable artificial intelligence systems. In: Proceedings of the 21st National Conference on Artificial Intelligence and the 18th Innovative Applications of Artificial Intelligence Conference, AAAI-06/IAAI-06. vol. 2, pp. 1766–1773 (11 2006)
6. Dockhorn, A., Apeldoorn, D.: Forward Model Approximation for General Video Game Learning. In: 2018 IEEE Conference on Computational Intelligence and Games (CIG) (2018), to be published
7. Dockhorn, A., Tippelt, T., Kruse, R.: Model Decomposition for Forward Model Approximation. In: 2018 IEEE Symposium Series on Computational Intelligence (SSCI) (2018), to be published
8. Giryes, R., Elad, M.: Reinforcement Learning: A Survey. Eur. Signal Process. Conf. pp. 1475 – 1479 (2011)
9. Guyon, I., Elisseeff, A.: An Introduction to Variable and Feature Selection. Journal of Machine Learning Research (JMLR) 3(3), 1157–1182 (2003)
10. Ilhan, E., Etaner-Uyar, A.S.: Monte Carlo tree search with temporal-difference learning for general video game playing. In: 2017 IEEE Conference on Computational Intelligence and Games (CIG). pp. 317–324. IEEE, New York (aug 2017)
11. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet Classification with Deep Convolutional Neural Networks. Advances In Neural Information Processing Systems pp. 1–9 (2012)
12. Kruse, R., Borgelt, C., Braune, C., Mostaghim, S., Steinbrecher, M.: Computational Intelligence. Texts in Computer Science, Springer London, London, 2nd editio edn. (2016)
13. Laurent, A., Marsala, C., Bouchon-Meunier, B.: Improvement of the Interpretability of Fuzzy Rule Based Systems: Quantifiers, Similarities and Aggregators. In: Lawry, J., Shanahan, J., L. Ralescu, A. (eds.) Modelling with Words: Learning, Fusion, and Reasoning within a Formal Linguistic Represntation Framework, pp. 102–123. Springer Berlin Heidelberg (2003)
14. Marsala, C.: Application of fuzzy rule induction to data mining. In: Andreasen, T., Christiansen, H., Larsen, H.L. (eds.) Flexible Query Answering Systems. pp. 260–271. Springer Berlin Heidelberg, Berlin, Heidelberg (1998)
15. Marsala, C., Bouchon-Meunier, B.: Fuzzy data mining and management of interpretable and subjective information. Fuzzy Sets and Systems 281, 252–259 (dec 2015)
16. Mellouli, N., Bouchon-Meunier, B.: Abductive reasoning and measures of similitude in the presence of fuzzy rules. Fuzzy Sets and Systems 137(1 SPEC.), 177–188 (2003)
17. Mitra, S., Konwar, K., Pal, S.: Fuzzy decision tree, linguistic rules and fuzzy knowledge-based network: generation and evaluation. IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews) 32(4), 328–339 (nov 2002)
18. Pan, F., Cong, G., Tung, A.K.H., Yang, J., Zaki, M.J.: Carpenter. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '03. p. 637. ACM Press, New York, New York, USA (2003)
19. Pang, B., Lee, L.: Opinion Mining and Sentiment Analysis. Foundations and Trends® in Information Retrieval 2(1–2), 1–135 (2008)
20. Pei, J., Tung, A.K., Han, J.: Fault-tolerant frequent pattern mining: Problems and challenges. DMKD 1, 42 (2001)

21. Perez-Liebana, D., Samothrakis, S., Togelius, J., Schaul, T., Lucas, S.M., Couetoux, A., Lee, J., Lim, C.U., Thompson, T.: The 2014 General Video Game Playing Competition. IEEE Transactions on Computational Intelligence and AI in Games 8(3), 229–243 (2016)
22. Perez-Liebana, D., Stephenson, M., Gaina, R.D., Renz, J., Lucas, S.M.: Introducing real world physics and macro-actions to general video game ai. In: 2017 IEEE Conference on Computational Intelligence and Games (CIG). pp. 248–255. IEEE (2017)
23. Quinlan, J.R.: Induction of Decision Trees. Machine Learning 1(1), 81–106 (1986)
24. Rogers, S.: Level Up!: The Guide to Great Video Game Design. Wiley (2010)
25. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) 115(3), 211–252 (2015)
26. Schaul, T.: An extensible description language for video games. IEEE Transactions on Computational Intelligence and AI in Games 6(4), 325–331 (2014)
27. Schmidhuber, J.: Deep Learning in neural networks: An overview. Neural Networks 61, 85–117 (2015)
28. Silver, D., Huang, A., Maddison, C.J., 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.: Mastering the game of Go with deep neural networks and tree search. Nature 529(7587), 484–489 (2016)
29. Sweetser, P., Wyeth, P.: GameFlow: A Model for Evaluating Player Enjoyment in Games. Comput. Entertain. 3(3), 3–3 (2005)
30. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I.J., Fergus, R.: Intriguing properties of neural networks. CoRR abs/1312.6199 (2013)