Chess Revision: Acquiring the Rules of Chess Variants through FOL Theory Revision from Examples

Stephen Muggleton[1*], Aline Paes[12**], Vítor Santos Costa[3***] and Gerson Zaverucha[2**]

[1] Department of Computing, Imperial College London, UK
shm@doc.ic.ac.uk
[2] Department of Systems Eng. and Computer Science, UFRJ, Brazil
{ampaes, gerson}@cos.ufrj.br
[3] CRACS and DCC/FCUP, Universidade do Porto, Portugal
vsc@dcc.fc.up.pt

Abstract. The game of chess has been a major testbed for research in artificial intelligence since it requires focus on intelligent reasoning. Particularly, several challenges arise to machine learning systems when inducing a model describing the rules of chess, including the creation and definition of the examples, the learning of a model which correctly represents the official rules of the game, covering all the branches and restrictions of the correct moves, and the comprehensibility of such a model. Besides, the game of chess has inspired the creation of numerous variants, ranging from faster to more challenging or to regional versions of the game. The question arises if it is possible to take advantage of an initial classifier of chess as a starting point to obtain classifiers for the different variants. We approach this problem as an instance of theory revision from examples. The initial classifier of chess is inspired by a FOL theory approved by a chess expert and the examples are defined as sequences of moves within a game. Starting from a standard revision system, we argue that abduction and negation are also required to best address this problem. Experimental results show the effectiveness of our approach.

1 Introduction

Game playing is a fundamental human activity, and has been a major topic of interest in AI communities since the very beginning of the area. Good performance in games often requires a significant amount of reasoning, making this area one of the best ways of testing computational schemes of intelligent systems. Particularly, the game of chess has been sometimes referred as the Drosophila of AI, since it has offered several challenges to the area, mainly because any

* Research Chair in Machine Learning founded by the Royal Academy of Engineering and Microsoft Research
** Financially supported by CNPq
*** Financially supported by Fundação para a Ciência e Tecnologia

Presented at the 19th International Conference on Inductive Logic Programming (ILP'09).

application involving chess must focus on intelligent reasoning. Datasets based on chess have been created and used as testbeds for machine learning systems throughout several decades, to learn a classifier of the game or playing strategies [3]. In order to acquire a meaningful representation of the classifier of the game, one could take advantage of the expressiveness of first-order logic and use *Inductive Logic Programming (ILP)* [7] methods to induce the game's rules written as a logic program, from a set of positive and negative examples and background knowledge (BK). Previous work has demonstrated the feasibility of using ILP to acquire a rule-based description of the rules of chess [4] using structured induction.

On the other hand, chess has inspired the creation of several chess based games, to be more challenging to the player, to produce an easier and faster variant of the original game, or still to be a regional version of the game, originating a *variant* or a *new version* of the game. There are numerous chess variants, which can be defined as any game that is derived from, related to or inspired by chess, where the capture of the enemy king is the primary objective [9]. For instance, the *Shogi* game, is the most popular Japanese version of Chess. Although both games have similar rules and goal, they also have essential differences. For example, in Shogi a captured piece may change sides and return to the board [4].

As the acquisition of knowledge is a difficult task, time consuming and error prone, one could take advantage of a classifier of chess as a starting point to obtain the rules of a variant of the game. Such rules contain useful information to the variant in respect to the similarities between both games. However, they need to be modified to represent the particular aspects of the variant. Modifying a set of rules so that they could explain a new set of examples is the task of *Theory Revision from Examples* [12]. Thus, in this work we handle the problem of obtaining the rules of variants of chess from the rules of chess as an instance of Theory Revision from Examples. This task is also closely related to *shallow Transfer Learning* [11], whose task is generalizing to different variations of the same domain. To perform such a task we started from the current version of FORTE revision system [10, 1] and show that abduction and negation should be introduced in the revision process to best address the problem. We start by discussing the revision system, emphasizing the modifications performed on it in section 2. Next, we briefly describe the format we define for the examples and the initial knowledge in section 3. We present experimental results in section 4 and finally we conclude the paper and discuss future work in section 5.

2 Chess Revision

ILP systems learn using a set of examples and background knowledge (BK) assumed as correct. On the other hand, theory revision (TR) from examples consider that the BK could also contain incorrect rules, which, after being modified, would better reflect the dataset. Thus, in TR the BK is divided into two sets: one containing the rules subject to modification, named here as *initial theory* and

[4] It is suggested that this innovative drop rule was inspired by the practice of 16th century mercenaries who switched loyalties when captured [9].

the other composed of rules known to be correct and therefore not modifiable, containing intensional definitions of the fundamental relations used to define a domain, named as *fundamental domain theory (FDT)* [10]. The goal of theory revision is to identify points in the initial theory preventing it from correctly classifying some example, and propose modifications to such points, so that the revised theory together with the FDT is correct. The first step in a revision process is to find the clauses and/or antecedents responsible for the misclassification of some example, the *revision points.* After finding the revision points, any revision system must propose modifications to them, through *generalization and specialization revision operators* [12].

In this work we follow the new version of FORTE revision system [10] as modified in [1] to allow the use of bottom clause and modes declarations [5]. In order to best address the revision of the rules of Chess, we performed three main modifications on the current version of the system, described as follows.

Starting the revision process by deletion of rules. In an attempt to decrease the complexity of the theory and consequently of the whole revision process, we introduced a first step of deletion of rules. This process is performed as a hill-climbing iterative procedure, where at each iteration the clauses used in proofs of negative examples are selected, the deletion of each one is scored and the one improving the score at most is chosen. This step restarts from the modified theory and finishes when no deletion is able to improve the score. This procedure both reduces theory size and noise.

Using abduction during the revision process. Abduction is concerned about finding explanations for observed facts, viewed as missing premises in an argument, from available knowledge deriving those facts [2]. Usually, theory revision systems, including FORTE, use abduction when searching for generalization revision points, to locate faults in a theory and suggest repairs to it, determining a set of assumptions that would allow the positive example be proved [10]. We further benefit from abduction in two distinguished moments of the revision process. First, when searching for the revision points, we assume that faulting abducible predicates are true and continue to search for further revision points possibly depending on them. Then, such abducible predicates are included in the theory under revision, up to a maximum limit. The second further moment is when constructing a bottom clause for intermediate predicates. Those are predicates in the head of clauses but the dataset does not contain examples for them, since the examples are of the top-level predicate(s) only. However, to construct the bottom clause it is necessary to start from a positive example with the same predicate as the head of the clause being specialized. Thus, from a positive example proved by the current clause we obtain the required literal using the initial theory and FDT, similar to [6]. Next, we construct the bottom clause from such a literal and use it as search space for adding literals to the clause.

Using negated literals in the theory. FORTE was neither able to introduce negated literals in the body of the clause nor revise negated antecedents. Negation is essential to elegantly model the chess problem, since we need to represent concepts such as *the king is* **not** *in check*, among others. In order to add negated literals in the body of the clause, we allow the bottom clause procedure construction to generate negated literals, requiring either such a literal has only input variables or its output variables have not used again. We also introduced a procedure for handling a faulty negated literal during the revision process. Roughly speaking, if the negated literal is responsible for a failed proof of positive examples, it is treated as a specialization revision point. On the other hand, if the negated literal takes part in a proof of a negative example, it is treated as a generalization revision point.

3 The Examples and the Background Knowledge

Examples. In our framework, the legal moves in the chess variant are the positive examples and the illegal moves the negative examples. The dataset is composed of a set of simulated games, where each game is limited to a specified maximum number of legal moves, considering both players. The moves are within a game, aiming to represent castling and en-passant, which require the history of the game (p.ex., the rook and the king involved in a castling must have not moved yet in the whole game), and promotion, which requires updating the board to represent the promoted piece. Each move is a ground fact, whose terms are the number of the move, the current and next status of the piece. For example, $move(9, pawn, white, c, 7, rook, white, c, 8)$ states that in the $9th$ move executed in the game a *white pawn* moves from $c, 7$ to $c, 8$ and is promoted to a *rook*. Each simulated game, besides the examples, has a set of ground facts, representing the position of the pieces on the board and the pieces removed from the games. Considering the example above, the set of facts would contain $board(10, rook, white, c, 8)$, stating that after the $9th$ move, there is a *white rook* in the position $c, 8$ on the board and $out\_board(10, pawn, white, 0, 0)$, giving the information that a *white pawn* was promoted in the $9th$ move of the game. The board setting is updated according to the last legal move performed. A move generator procedure is responsible for creating the dataset of simulated games.

Background Knowledge. In the chess revision problem, the initial theory describes the rules of the standard game of chess, which will be revised using the set of examples for its variant. The theory encompass all the possible branches of a move, considering the official rules of chess. For example, *in case* the king is under attack, a piece *must* only move to stop the attack, by *either* capturing the attacking piece *or* moving to the way between the king and the attacking piece. This theory is inspired by the one learned in [4] using structured ILP and Progol [5].[5] The major differences between the theory used in the present work and the previous one are the clauses describing castling, en-passant and promotion, since the authors of that work opted to not represent any such special move. The FDT contains fundamental rules to the problem of chess in general, such as to calculate the difference between two positions on a board. The initial theory has 109 clauses and the FDT 42 clauses. The work in [4] had 61 clauses in the BK and it learned 61 clauses.

[5] The resulting theory was approved by Professor Donald Michie, who could be considered a chess expert.

4 Experimental Results

Experimental methodology To experiment the proposal of the paper, we generated datasets with 5 simulated games where each stage of the game has 1 positive and 5 negative examples and the maximum number of legal moves is 20, for 3 different chess variants. We performed 5-fold cross validation and scored the revisions using f-measure. We proceed with this section by describing the chess variant followed by the revisions performed to obtain its theory. The variants include a smaller version of chess, a version with an unusual rule and a variant with larger board and unusual pieces.

Using smaller boards: Gardner's Minichess. This is the smallest chess game (5X5) in which all original chess pieces and legal moves are still used, including pawn double move, castling and en-passant [9]. The revisions are as follows.

1. The delete rule step was able to remove the following clauses from the theory: file(f). file(g). file(h). rank(6). rank(7). rank(8). promotion_zone(pawn,white,File,7).
2. The add rule generalization operator created the following clause: promotion_zone(pawn,white,File,5) (white pawn promoting in rank 5).

The final accuracy was 100% and the returned theory matches the rules of Gardner Minichess.

Unusual rule: Free-capture chess. This variant of chess allows a piece to capture friendly pieces, except for the friendly king, besides the opponent pieces [9].

1. The chess theory makes sure that a piece when attacking does not land on a friendly piece, requiring that the colours of them both are different. The revision created a new clause by first deleting the predicate requiring the colours are different and then adding a literal restricting the attacked pieces to be different from the king. Note that the original rule is kept on the theory.
2. There is a specific rule in the case of a king attacking, since the restrictions on the king's move must be observed. The revision system deleted the literal requiring the colour of a piece attacked by the king be different from the colour of the king.

The final accuracy was 100% and we can say that the returned theory perfectly corresponds to the target theory.

Unusual pieces and larger board: Neunerschach. This is a chess variant played on a 9x9 board. There is a piece called as *Marschall* replacing the Queen and moving like it. The extra piece is the *Hausfrau*, which moves as a Queen but only two squares [9]. The theory was revised as follows.

1. The delete rule step removed the clause defining the queen as a piece;
2. The abduction procedure included facts defining the $marschall$ and $hausfrau$ as pieces on the theory;
3. From the rule defining the basic move of the queen, the add rule operator created a rule for the $Marschall$.
4. New rules were added to the theory, defining the basic move of $Hausfrau$ and introducing the facts $file(i)$ and $rank(9)$.

Since in this dataset no promotion moves were generated, due to the size of the board, the revision process failed on correcting the promotion on the last rank. We expect that using games with a larger number of total moves will allow us to represent such a promotion. Nevertheless, the final accuracy was 100%.

5 Conclusions

We presented a framework for applying the knowledge learned to the rules of chess to learn variants of chess through theory revision and a set of generated examples. We described the modifications performed on the revision system, including the introduction of an initial step for deleting rules, the use of abduction and negation. Three variants of chess were experimented and the revision was able to return final theories correctly describing most of the rules of the variants. The missing case was related to the promotion in the last rank of a large board.

In order to decrease the runtime of the revision process we intend to use the stochastic local search algorithms developed in [8]. We would like to try induce the variants of Chess using standard ILP system, such as Progol [5], with the chess theory as BK. Beforehand, it is expected that these systems are not successfull in the cases requiring specialization, since usually they do not perform such operation. If the chess theory is not used as BK, we would not take advantage of the initial knowledge about the domain.

Additionally, we want to take a further step towards the acquirement of more complex chess variants, such as the regional chess games *Shogi* and *Xiangqi*. We invite the ILP community to experiment the datasets developed in this work in order to induce the rules of chess and variants of chess. They will become publicly available after the ILP conference.

References

1. Ana Luísa Duboc, Aline Paes, and Gerson Zaverucha. Using the Bottom Clause and Modes Declarations on FOL Theory Revision from Examples. In *Proc. of the 18th Int. Conf. on ILP*, LNAI, 5194, pages 91–106. Springer, 2008.
2. Peter Flach and Antonis Kakas. *Abduction and Induction: Essays on their Relation and Integration.* Kluwer Academic Publishers, 2000.
3. Johannes Fürnkranz. Recent Advances in Machine Learning and Game Playing. *OGAI-Journal*, 26(2):147–161, 2007.
4. John Goodacre. Master thesis, Inductive Learning of Chess Rules Using Progol. Programming Research Group, Oxford University, 1996.
5. Stephen Muggleton. Inverse Entailment and Progol. *New Generation Computing*, 13(3&4):245–286, 1995.
6. Stephen Muggleton and Christopher H. Bryant. Theory completion using inverse entailment. In *Proc. of the 10th Int. Conf. on ILP*, LNAI,1866, pages 130–146. Springer, 2000.
7. Stephen Muggleton and Luc De Raedt. Inductive Logic Programming: Theory and Methods. *J. Log. Program.*, 19/20:629–679, 1994.
8. Aline Paes, Gerson Zaverucha, and Vítor Santos Costa. Revising FOL Theories from Examples through Stochastic Local Search. In *Proc. of the 17th Int. Conf. on ILP*, LNAI, 4984, pages 200–210. Springer, 2007.
9. David B. Pritchard. *The Classified Encyclopedia of Chess Variants.* John Beasley, 2007.
10. Bradley L. Richards and Raymond J. Mooney. Automated Refinement of First-order Horn-Clause Domain Theories. *Machine Learning*, 19(2):95–131, 1995.
11. Sebastian Thrun. Is Learning the n-th Thing any Easier than Learning the First? In *Adv. in Neural Inf. Proc. Systems 8, NIPS*, pages 640–646. MIT Press, 1995.
12. Stefan Wrobel. First-order theory refinement. In L. De Raedt, editor, *Advances in Inductive Logic Programming*, pages 14–33. IOS Press, 1996.