Enhancing Artificial Intelligence in Games by Learning the Opponent's Playing Style

Fabio Aiolli and Claudio Enrico Palazzi

Abstract As virtual environments are becoming graphically nearly realistic, the need for a satisfying Artificial Intelligence (AI) is perceived as more and more important by game players. In particular, what players have to face nowadays in terms of AI is not far from what was available at the beginning of the video games era. Even nowadays, the AI of almost all games is based on a finite set of actions/reactions whose sequence can be easily predicted by expert players. As a result, the game soon becomes too obvious to still be fun. Instead, machine learning techniques could be employed to classify a player's behavior and consequently adapt the game's AI; the competition against the AI would become more stimulant and the fun of the game would last longer. To this aim, we consider a game where both the player and the AI have a limited information about the current game state and where it is part of the game to guess the information hidden by the opponent. We demonstrate how machine learning techniques could be easily implemented in this context to improve the AI by making it adaptive with respect to the strategy of a specific player.

1 Introduction

Games embody one of the main revenue sources for the digital entertainment industry, attracting every day a multitude of new customers and astonishing them with tremendous advancements in terms of graphics quality. Indeed, the current 3D graphics is far away from the flashing pixels of the first video games. Unfortunately,

Fabio Aiolli
Pure and Applied Math Department, University of Padova, Via Trieste 63 - 35131, Padova, Italy, e-mail: aiolli@math.unipd.it

Claudio Enrico Palazzi
Pure and Applied Math Department, University of Padova, Via Trieste 63 - 35131, Padova, Italy, e-mail: cpalazzi@math.unipd.it

the same level of advancement did not happen in another very important aspect of gaming: the Artificial Intelligence (AI) that commands the virtual opponent of a player. Current games still rely on decision trees with almost deterministic sequences of actions and reactions. Standard difficulty degrees are generally offered to provide players with harder virtual opponents. Yet, these degrees are generally associated with just i) deeper levels that the AI can access in the decision tree rather than adapting its decisions to the human opponent, or ii) an improvement of the quality of the virtual opponent's features (e.g., speed, armor, weapons) rather than improving its ability in using them. Expert players can hence quickly find weak spots in the AI, adopting a routine of actions that (almost) always leads to victory; at the same time, the game quickly gets boring.

Viceversa, part of the fun in playing against another human opponent relies on the fact that she/he can uncover possible winning tactics and adapt to them, thus prolonging the challenge and the fun considerably. Providing this adaptation ability also to the AI is going to become a crucial feature of future game releases, determining their market success. Since its importance, we show how adaptation in game's intelligence can be generated through machine learning techniques that model a human's behavior.

Furthermore, classic searching techniques may not be feasible for certain games. Indeed, games can be classified into two main categories depending on whether participants have or not a complete knowledge of the game state at any moment. Typical exemplars of the two classes are represented by Chess and Poker, respectively. When players have just a limited knowledge of the game state, resorting to traditional search in decision trees may result in an AI as effective as a random decision maker. Instead, machine learning techniques could be exploited to mimic the humans's ability in intuiting the opponent's intentions after several game sessions.

As a case study for this subject, we consider *Ghosts*[1], a simple board game played by two opponents and that had not yet a computer based version. The game is particularly interesting for our study as players do not have a complete knowledge of the game state: they can both see the position of game pieces on the board, but they cannot see the type of the opponent's ones. Depending on this information, different tactics would be adopted (i.e., attack the opponent's piece, leave it alone, run away from it). Therefore, in order to win, a player has also to infer the type of each of the opponent's pieces. This information can be extracted from the player's behavior, also keeping in mind that different players can adopt different strategies, for instance, by resorting more or less frequently to bluffing.

In this work we stepped through different phases. First, we have generated a computer based version of *Ghosts* that can be played both against an AI and against another human opponent. Second, we have collected and analyzed tens of game sessions to extract behavior features. Third, we have endowed the AI with machine learning capabilities so as to associate the behavior features of a specific player with a presumed type of a piece. In essence, by observing how a player acts in different

[1] The board game has been invented by Alex Randolph and is sold in Germany by *Drei Magier Spiele*. Its original (German) name is: *Die guten und die bösen Geister*, i.e., good and bad ghosts; for brevity, we simply name it *Ghosts*.

game state configurations, the AI becomes able to classify tactics employed by that specific player and to adapt to them. Finally, we have tested the system, proving its ability in profiling the player's strategy and adapting to it.

The rest of the paper is organized as follows. In Section 2 we review some background in game-related machine learning and describe the specific game we have considered as a case study. The machine learning capabilities we devised to improve the game AI is presented in Section 3. Section 4 describes the experimental scenario and reports the corresponding outcomes. Finally, in Section 5 conclusions and future directions for this research work are provided.

2 Background

Delving into the Internet, the first video game seems to be a simple tennis-for-two created by Higginbotham in 1958 to entertain visitors of the Brookhaven National Laboratory, a US nuclear research lab in Upton, New York. Only one year later, Samuel proposed the first self-learning gaming program, i.e., Checkers, that represented a very early demonstration of the fundamental concept of AI in games [12]. Nowadays, all video games include some AI that may act as a virtual opponent or as a component of the game itself. Yet, the AI of current games show only little advancements if compared to its ancestors; only for few specific games the AI has achieved great improvements (e.g., Chess [6]).

In general, games can be categorized into two main classes: games where the players possess perfect information about the current game state (e.g., Chess, Tic-tac-toe) and games where players can rely only on imperfect information (e.g., Poker, Rock-paper-scissors). In the former case, all the information related to a certain game state are known by players; whereas in the latter, players may not be aware of some information such as the opponent's cards or the placement of the opponent's pieces on the board.

The AI of *perfect information games* can easily evaluate a given game state by just searching all possible continuations to a fixed depth. For this kind of games, the main problem in developing an AI is related to the capability of pre-computing correct evaluations of each game state and then storing and retrieving them in an efficient manner [1, 9, 5].

Instead, with *imperfect information games*, deep search may not be feasible and storing pre-computed evaluations may not result in significant improvements in the playing strength [3, 8]. In these case, techniques like temporal difference learning are also unsuitable as the intermediate states of a game are only partially determined [11]. Alternative solutions have hence to be explored to enhance the level of the AI. For instance, *simulation search* [13, 2, 10, 14] evaluates the possible next moves by self-playing a multitude of simulated game sessions, considering the current state as the starting point and utilizing different values for the indeterministic parameters (i.e., dice rolls, cards held by the opponent player, etc.). To generate real-time responses during the game, these simulations can be run before the game and statistics

can be stored to be promptly available during the game. Unfortunately, the branching factor of certain games may considerably limit the effectiveness of this technique.

Instead, we propose a new machine learning approach that mimics the human's ability in evaluating important information about the current game state that goes beyond, for instance, the board position. In essence, our mechanism models the opponent's behavior over several game sessions so as to be able to exploit the weaknesses and the repetitive behaviors of the considered human player.

2.1 A Representative Case Study: Ghosts

For our experiments, we need a simple, yet representative exemplar of imperfect information game. *Ghosts* embodies a perfect case as, not only it belongs to this class of games, but it is also governed by few simple rules, which follow. Two players have to place 8 ghosts each at the back of a 6x6 board as shown in Fig. 1. Each player has 4 good ghosts and 4 bad ghosts, but the information about which are good and which bad is hidden to the opponent player. Each turn a player moves one of her/his ghosts one square vertically or horizontally; if by doing so the ghost is moved onto an opponent's ghost, the latter is captured by the former. In order to win, different possibilities are available to a player: i) having all of her/his bad ghosts captured by the opponent player, ii) capturing all the good ghosts of the opponent player, iii) moving one of her/his good ghosts off the board from one of the opponent's corner squares. Clearly, one of the interesting aspects of *Ghosts* is its bluffing element which is differently utilized by different human players.

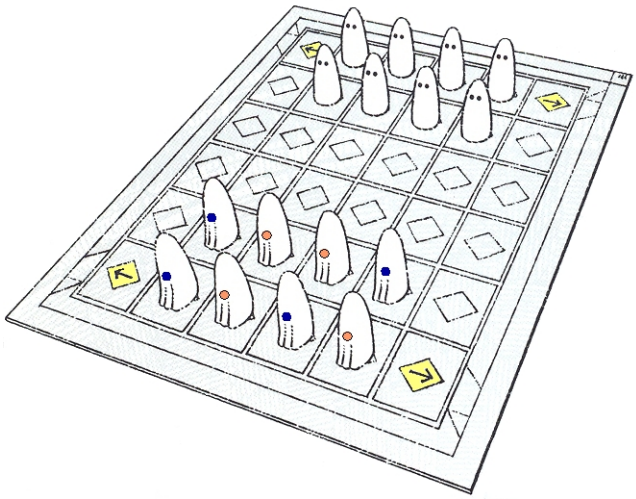

Fig. 1 Initial set-up of *Ghosts* (the image is taken from the manual of the board game).

3 Enhancing the AI with Player Profiling Capabilities

The most interesting problems in game AI are typically grouped with respect to certain characteristics. One of the most intriguing characteristics is how much information is available to the players. This leads us to distinguish between perfect information games and imperfect information games as discussed in Section 2. Clearly, the latter is more interesting as it represents a more challenging case.

Ghosts embodies a good case study as it falls in the imperfect information games class, yet, it simple enough to be analyzed. In *Ghosts* a player does not have any information about the type of the opponent's ghosts (i.e., good or bad); thereby, any search state space based technique, e.g. min-max algorithms, will fail miserably. In other words, without any other heuristic judgement, the behavior of a machine driven player could not be better than a any trivial random player.

Indeed, in order to plan its moves, an AI algorithm would certainly benefit from some other source of information about the type of the opponent's ghosts (the missing information). In this work, we propose to get this additional information from the playing style of a player. The basic assumption we make is that different players have different playing behaviors (being aggressive, bluffing, etc.) and they tend to move pieces of a certain type in a similar way when facing similar game situations. Specifically, our claim is that knowing the playing style of a player, it is possible to recognize the type of a given ghost by its moves. We can then use a machine learning algorithm [11] to compile a behavior profile of good and bad pieces of a player. During future game sessions, this knowledge can be used to predict the type of a ghost in the board and possibly to define higher level game heuristics, like min-max algorithms, based on this predictions.

The machine learning methodology we have used is very simple and it is based on a prototype-based algorithm. Specifically, for each player, a prototype of good and bad ghosts is trained based on 17 features which have been considered informative to determine the nature of a ghost in the game.

In particular, the following features have been chosen: 8 features with binary values representing which was the initial position of the piece on the board among the eight possible ones, 5 features representing the moves of the piece during the game session (if it is the first piece that the player moved, if it is the second piece that the player moved, the number of backward, forward, and lateral moves already performed by that piece), and the remaining 4 features representing the piece's behavior when it has been under threat of being captured (how many times it has reacted by capturing the opponent piece, how many times it has escaped by moving to another board position, how many times it has remained on its position, and how many times it has moved from its square to threat another opponent's piece).

To build the prototype for a player, our algorithm needs first to collect data, i.e., the training set, from previous game sessions with the same player. For each of these sessions and for each piece a corresponding feature vector is built according to the criteria above which are based on the behavior of the piece in the game. The prototype for good (or bad) pieces is then determined as the average among the feature vectors representing good (or bad) pieces. More formally, given $G = \{g_1, \ldots, g_n\}$

the set of feature vectors for good pieces of a given player, and $B = \{b_1, \ldots, b_n\}$ the set of available feature vectors for bad pieces of the same player, the prototype vectors are computed in the following way:

$$P_G = \frac{1}{n}\sum_{i=1}^{n} g_i, \text{ and } P_B = \frac{1}{n}\sum_{i=1}^{n} b_i. \tag{1}$$

Now, let be given a new feature vector f representing the profile of a piece of unknown type on the board; a badness score can hence be computed by using the normalized distance with respect to the player prototypes, i.e.,

$$s(f) = \frac{d(f,P_G) - d(f,P_B)}{d(f,P_G) + d(f,P_B)} \tag{2}$$

where $d(x,y)$ is the Euclidean distance between vectors x and y. Note that, the score is always a number between -1 (definitely good) and $+1$ (definitely bad).

On each move in the middle of a game session, the prediction of the type of the pieces on the board is performed in the following way. First, since the exact number of good and bad pieces (n_g and n_b, respectively) still on the board is a known information, a badness score for each of these pieces can be computed by utilizing (2). Then, the pieces are ranked based on this score and the n_b highest score pieces are predicted to be bad pieces.

The error committed in a prediction is computed as the number of bad pieces which are actually predicted as good ones. Needless to say, with the ranking method we used to discriminate between bad and good pieces, this error also corresponds to the number of good pieces which are incorrectly predicted as bad.

4 Experimental Results

In this section, we report on experimental results that show the effectiveness of our profiling procedure. To train our machine learning algorithm, we have generated a computer based version of *Ghosts* and collected a set of 81 game session logs for a player who was playing against other human players.

4.1 Profiling Evaluation in a Single Game Session

In this first set of experiments, we aim at studying how the prediction error decreases in a game as ghosts' profiles become more and more informative. This prediction improvement is naturally caused by the fact that some of the features for a piece can be still unavailable or underestimated at the very beginning of a game session, thus generating a bad estimate of the pieces' profiles. Yet, the precision of the predictor quickly improves as the game session continues. This represents a desirable property

as, in general, it is not so important to have a very low error when the game is at the very beginning, whereas it becomes crucial as the game session proceeds.

In particular, Fig. 2 plots the percentage of error that the machine learning algorithm performs during a single game. The same game experiment has been repeated five times (labeled in the chart as *g1*, *g2*, *g3*, *g4*, and *g5*, respectively): 50% of error means that two good (and two bad) ghosts over 4 were wrongly labeled as bad (good), 25% represents the case with only one good (and one bad) ghost wrongly labeled, and 0% is the when the system is providing perfect type prediction for all the ghosts. As expected, the error drops as the number of moves increases.

4.2 Profiling Evaluation with Varying Training Set Size

In a second set of experiments, the performance of the algorithm has been tested considering several game sessions against the same human player. To this aim, an estimate of the mean error in each game has been computed by using a leave-one-out procedure. This measure, very common in the machine learning community, was used as it provably gives a very good estimate of the error a learning algorithm will make on future games on average. Specifically, for each available game session

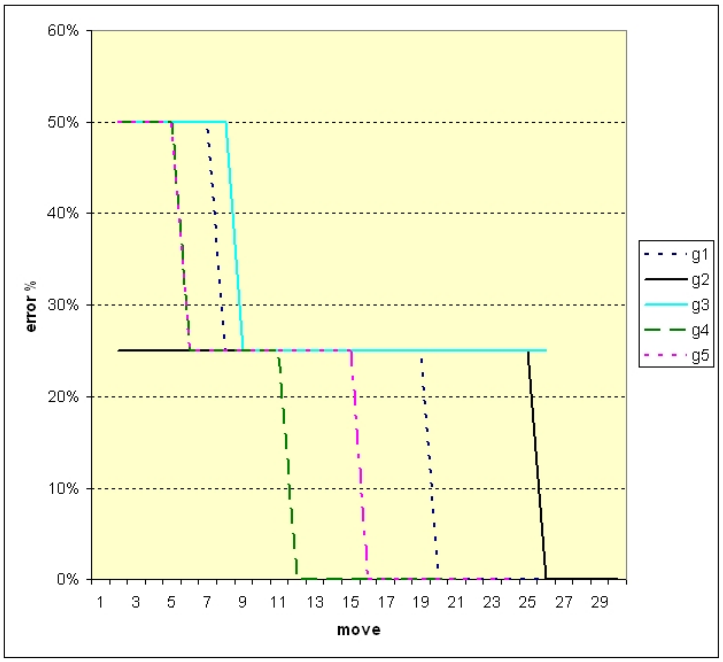

Fig. 2 Percentage of error performed by the machine learning algorithm when playing 5 games.

log, the mean error (the number of mistakes out of the number of guesses) has been computed using the model obtained by training on the remaining 80 game logs. All these results have been averaged to obtain a final estimation of the algorithm's performance. The leave-one-out estimation result has been 26.3% error.

Aimed at showing how the performance of the algorithm improves when employing higher numbers of training game logs, we have exploited the leave-one-out procedure also considering different training set sizes (i.e., 5, 10, 25, 50, 80). For each chosen size of the training log set, we have repeated the procedure each time considering a different left-out log; the averaged resulting error percentages are reported in Fig. 3.

As expected, with larger sizes of the training set, the machine learning algorithm is able to build more reliable profiles of the players, improving the performance of the prediction system. In particular, the error drops from 42%, when only 5 training logs are used, to 26.3% with 80 training logs.

5 Conclusion

AI is becoming a crucial, albeit still neglected aspect in games. Imperfect information games are particularly affected by the lack of smart virtual opponents thus

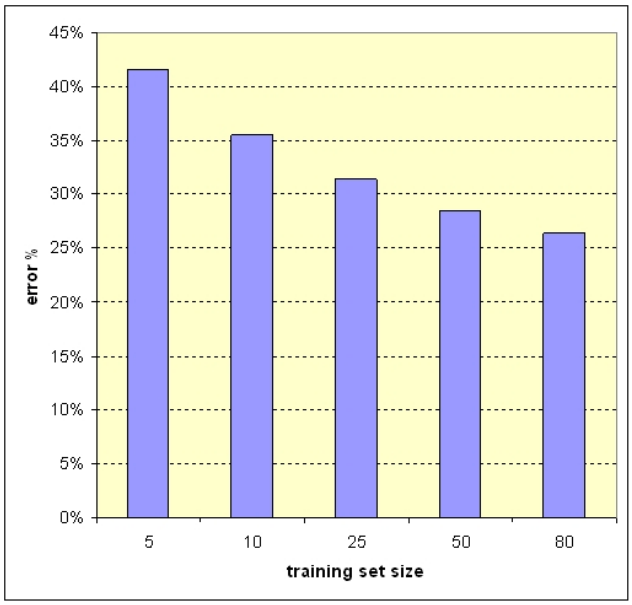

Fig. 3 Averaged leave-one-out error obtained by the machine learning algorithm varying the number of game logs used for training.

demanding new technical solutions. To this aim, we have designed and tested a new approach that improves the capability of the machine player by allowing it to adapt to its human opponent and exploit her/his weaknesses. In particular, through repeated game sessions, the AI models the behavior of a human player so as to be able to employ the strategy that best fits that player. Moreover, this profiling method can be easily plugged into any standard AI or temporal difference learning based algorithm to enhance their performance.

In order to test our solution, we have deployed a computer based version of *Ghost*, a simple, yet general, two-player imperfect information game. Results gathered during our experimental evaluation demonstrates how our approach may allow the AI to adapt to the player.

Our experiments have to be intended as proof of concept for the benefits that player profiling can produce when considering imperfect information games. We have hence used a very simple machine learning algorithm to test the viability of the general method. This work can hence be extended in several research directions. For instance, we intend to improve the methodology by using state-of-the-art machine learning algorithms such as, for instance, Support Vector Machines [15], and to enhance the set of features used to profile the opponent's behavior. Moreover, we also plan to apply our solution to more complex imperfect information games such as *Invisible Chess* and *Kriegspiel* which are heterodox Chess variation in which players are not informed of their opponents position and moves [4, 7].

Acknowledgement

Our deep gratitude goes to Ivan Mazzarelli and Federico Giardina for their technical contribution in developing the computer based version of the game and the experimental set-up.

References

1. Allis, V. (1988). A knowledge-based approach of Connect-Four the game is solved: White wins. Masters Thesis, Department of Mathematics and Computer Science, Vrije Universiteit, Amsterdam, Netherlands.
2. Billings, D., Pena L., Schaeffer, J., Szafron, D. (1999). Using probabilistic knowledge and simulation to play Poker. In Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99), Orlando, Florida, 697-703.
3. Billings, D. (2000). The first international RoShamBo programming competition. International Computer Games Association Journal 23(1), 42-50.
4. Bud, A., Albrecht, D., Nicholson, A., Zukerman, I. (2001). Playing Invisible Chess with Information-Theoretic Advisors. In Proc. 2001 AAAI Spring Symposium on Game Theoretic and Decision Theoretic Agents, California, USA, 6-15.
5. Buro, M. (1997). The Othello match of the year: Takeshi Murakami vs. Logistello. International Computer Chess Association Journal 20(3), 189-193.

6. Campbell, M. S. (1999). Knowledge discovery in Deep Blue. Communications of the ACM, 42(11), 65-67.
7. Ciancarini, P., Dalla Libera, F., Maran, F. (1997). Decision Making under Uncertainty: A Rational Approach to Kriegspiel. In J. van den Herik and J. Uiterwijk, editors, Advances in Computer Chess 8, 277-298.
8. Egnor, D. (2000). Iocaine powder. International Computer Games Association Journal 23(1), 33-35.
9. Gasser, R. (1995). Efficiently harnessing computational resources for exhaustive search. Ph. D. thesis, ETH, Zurich, Switzerland.
10. Ginsberg, M. L. (1999). GIB: Steps toward an expert-level Bridge-playing program. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-99), Stockholm, Sweden, 584-589.
11. Mitchell T. (1997). *Machine learning*. McGraw Hill.
12. Samuel, A. L. (1959). Some studies in machine learning using the game of Checkers. IBM Journal of Research and Development 3(3), 211-229.
13. Schaeffer, J. (2000). The games computers (and people) play. InM. V. Zelkowitz (Ed.), Advances in Computers, 50, 189-266.
14. Sheppard B. (1999). Mastering Scrabble. IEEE Intelligent Systems 14(6), 15-16.
15. Vapnik V. (1995). *The nature of statistical learning theory*. Springer-Verlag.