Player Skill Modeling in *Starcraft II*

Tetske Avontuur, Pieter Spronck, and Menno van Zaanen
Tilburg Center for Cognition and Communication
Tilburg University, The Netherlands

Abstract

Starcraft II is a popular real-time strategy (RTS) game, in which players compete with each other online. Based on their performance, the players are ranked in one of seven leagues. In our research, we aim at constructing a player model that is capable of predicting the league in which a player competes, using observations of their in-game behavior. Based on cognitive research and our knowledge of the game, we extracted from 1297 game replays a number of features that describe skill. After a preliminary test, we selected the SMO classifier to construct a player model, which achieved a weighted accuracy of 47.3% (SD = 2.2). This constitutes a significant improvement over the weighted baseline of 25.5% (SD = 1.1). We tested from what moment in the game it is possible to predict a player's skill, which we found is after about 2.5 minutes of gameplay, i.e., even before the players have confronted each other within the game. We conclude that our model can predict a player's skill early in the game.

Introduction

In competitive computer gaming, also called eSports, players of a game participate in tournaments to determine who has mastered the game the most. In real-time strategy (RTS) games, a popular game used for eSports is *Starcraft II* (Blizzard Entertainment, 2010). The online game divides players into 7 distinct leagues, based on their performance against other players. We therefore assume that a player's league is a good representation of their level of skill.

Skill differences between novices and experts have been researched in several domains. Studies in chess suggest that because of their extensive knowledge, experts are strong at recognizing patterns, make quick decisions based on observed patterns, and are able to make effective general assumptions based on chunks of information (Gobet and Simon 1998). Research with airplane pilots and athletes shows that experts see related cues faster, make fewer mistakes, and pay less attention to unrelated cues than novices (Schriver et al. 2008; Chaddock et al. 2011).

To examine how the differences between *Starcraft II* players of different skill levels affect gameplay, we created a player model focused on skills (van den Herik, Donkers, and Spronck 2005). Our goal is to accurately distinguish, during gameplay, the leagues to which *Starcraft II* players are assigned. We are particularly interested in determining the player's league as early in the game as possible, to allow an AI opponent to adapt its tactics and strategies to the player's level before in-game confrontations with the player have taken place.

Background

We discuss being an expert in chess playing, and compare this to being an expert in video games. We then discuss player modeling in general, and describe *Starcraft II* and our reasons for choosing the game for the present research.

Expertise in chess playing

With respect to the thought processes of chess players, research shows no difference in depth of search between chess masters and grandmasters, although the latter produce move of higher quality and are better at remembering non-random board configurations (Campitelli et al. 2007; Reingold et al. 2001).

Chase and Simon (1973) found that expert chess players were better at remembering meaningful board configurations, while there was no difference in performance between experts and novices concerning meaningless configurations. Experts were also better at reproducing the position of a chess piece after viewing it for five seconds. This research lead to the chunking theory: the idea that experts make decisions based on a large number of chunks of information that are stored in their long term memory, helping them to recognize patterns and make quicker decisions.

Gobet and Simon (1998) extended this theory into the template theory where a set of chunks forms a complex structure in memory, which allows grandmasters to memorize relevant information, recognize board positions, and consequently make fast decisions. Gobet and Simon (1996) also studied grandmasters playing six opponents simultaneously. They found that grandmasters reduce the search space by using recognition patterns, based on their extensive knowledge of the game and their opponents. Campitelli and Gobet (2004) confirmed their findings.

Expertise in video games

An important difference between chess and modern video games is 'pacing' (the number of actions per time unit), which generally is much higher and frantic for video games. Green and Bavelier (2006; 2007) examined the difference in the allocation of attention between video gamers and non-gamers. They conducted a functional field-of-view task to measure how well a person can locate a central task whilst being distracted by a number of visible elements and one other central task. The gamers were better able to enumerate and track several simultaneously-moving stimuli over time. They also had better visuospatial attention, allowing them to effectively localize one or two central tasks among a number of distractions (Green and Bavelier 2007). When non-gamers were asked to play an action game for 10 hours, they showed significant improvement in attentional resources and visuospatial attention. In other words, experience with a game seems to improve the players' ability to multi-task. Dye, Green, and Bavelier (2009) examined the allocation of attention to a number of alerting cues. They found that action gamers responded quicker to such events and attended to them more accurately than non-gamers.

Player modeling

Bakkes, Spronck, and van Lankveld (2012) define a player model as an abstracted description of a player in a game environment. A player model may encompass characteristics such as preferences, strategies, strengths, weaknesses, and skills (van den Herik, Donkers, and Spronck 2005).

Research into player models for classic board games originated with Carmel and Markovitch (1993) and Iida et al. (1993). The main goal is to create strong game-playing artificial intelligence (AI). Such research provides information on how to model expert opponents in games, and to create AI that imitates experts. van der Werf et al. (2002) focused on predicting moves in the game of Go by observing human expert play. Kocsis et al. (2002) used a neural network to predict the best moves in Go from patterns in training data consisting of expert human play.

In video games, research into player modeling also focuses on increasing the effectiveness of artificial players. In this case 'effectiveness' does not necessarily refer to 'being a strong player'; rather, it often concerns raising the entertainment value of the game by providing the human player with an opponent which is a good match for their skills (van den Herik, Donkers, and Spronck 2005). Schadd, Bakkes, and Spronck (2007) examined player modeling in the real-time strategy game *Spring*. They were able to successfully classify the strategy of a player using hierarchical opponent models. Drachen, Canossa, and Yannakakis (2009) collected data from 1365 *Tomb Raider: Underworld* players. They used a self-organizing map as an unsupervised learning technique to categorize players into four types, and showed that it is possible to cluster player data based on patterns in gameplay. Weber and Mateas (2009) investigated the use of data mining techniques to model player strategies in the original *Starcraft* game.

Recently, researchers have attempted to construct player models based on psychological theories, looking into a player's profile rather than their in-game behavior. We give four examples. Yee (2006) modeled the motivational aspects of player behavior. Canossa (2009) modeled different playstyles as so-called "play-personas". Van Lankveld et al. (2011) correlated the results of the NEO-PI-R test with the gameplay behavior of 80 players of a *Neverwinter Nights* module. And recently, Tekofsky et al. (2013) sought to build a psychological profile of 13,000 *Battlefield 3* players based on their playstyle.

Starcraft II

The game used in this research is *Starcraft II: Wings of Liberty* (from hereon referred to as *Starcraft II*). It is a real-time strategy game where the players' goal is to destroy their enemy's base by developing their own base and an army. Players can choose from three different races to play, each of which plays very differently. To construct buildings and produce army units, a player needs minerals and gas. During the game, players unlock new options by constructing particular buildings.

To play the game well, the player must engage in both macro and micro-management. Macro management determines the economic strength of a player, represented by the construction of buildings, the gathering of resources and the composition of units. Micro-management determines how well a player is able to control small groups and individual units, including movements and attacks. A player's success depends heavily on the strategy followed. Strategic choices include finding a balance between building a strong economy and building a strong fighting force.

Using Blizzard's multiplayer system *Battle.net*, *Starcraft II* players compete against each other. There are four regions, each with their own ladder: Europe and Russia, North and Latin America, Korea and Taiwan, and South-East Asia. Games played on the ladder are ranked, and the *Battle.net* system automatically matches players of similar skill with each other. The average skill levels of the players in the four regions tend to differ, e.g., a player needs substantially stronger skills to gain a place on the top rung of the South-East Asian ladder than of the European ladder.

A ladder is divided into 7 leagues, which are (in order of increasing skill levels): bronze, silver, gold, platinum, diamond, master, and grandmaster. The bronze to platinum leagues each contain 20% of the population of players on the ladder. The diamond level contains 18%, and the master level contains almost 2%. The grandmaster level consists of the top 200 players of the ladder. Players always start out in the bronze league, and may be moved to one of the four bottom leagues after playing at least five placement matches. From that point on, they can gain or lose a rank on the ladder by winning or losing matches.

We have three reasons for choosing *Starcraft II* as our research environment: (i) there is a great degree of skill involved in playing the game; (ii) the ladder system provides a reliable and objective grouping of players who are roughly evenly skilled; and (iii) it is relatively easy to gather gameplay data as replays are available in large numbers via the Internet.

Experimental Setup

Our experimental setup consists of two parts. The first is construction of the dataset, i.e., the collection of data from

Table 1: General features.

Feature	Description
Player	Player ID
League	League in which the player is classified
Server	Server on which the game was played
Player race	Terran, Protoss, or Zerg
Opponent race	Terran, Protoss, or Zerg
Winner	Indicates whether the player won the game
Minute	Minute of game time, the first minute starts 90 seconds in the game

game replays and the extraction of relevant features. The second is the selection of the classifier that is used to build a player model from our dataset.

Data collection

For two months we collected game replay data from multiple websites[1]. The data collected was originally uploaded by players who posted their own games, or by tournament organizers. We stored data per player, only for 2-player games using version 1.4.2.20141 of *Starcraft II*. Therefore, every replay gave us access to information on two players, a winner and a loser. In total, we collected data on 1297 games, played between November 25, 2011, and February 13, 2012. We estimate that they comprise about 0.03% of all games played in that period (Avontuur 2012).

We only used games played on the American (63.3%) and European (36.7%) servers, as we assume that the skill levels of players in these regions are comparable (this assumption is based on personal experience rather than hard data). The data covered results from 1590 different players. We created a dataset of 49,908 instances, each describing one player's behavior during a time slice of one minute of game time. The instances were distributed over the leagues as follows: bronze 4082, silver 3979, gold 5195, platinum 8066, diamond 10,088, master 12,747, and grandmaster 5751 instances (Avontuur 2012). From this distribution it is clear that our dataset is skewed in favor of more experienced players. This is not unexpected, as experienced players are more likely to be involved with the *Starcraft II* community, and their games are of more interest to others. We used the `sc2reader` Python library and the `sc2gears` program to extract game and player information from the replay files. They provided us with an identification of the player, and a list of in-game actions that the players performed.

The feature set that was stored for each instance in the dataset contains three parts. The first part consists of general information on the game and the player, as described in Table 1. The second part is 'per-minute' data on the minute of game time that is represented by the instance. The third part is 'over-all' data on the whole game up to and including the represented minute. The second and third part are described in Tables 2, 3, 4, and 5 (further detailed below). From the feature set we excluded all actions that happened during the first 90 seconds of game time, as those concern the starting-up phase in which not much happens.

[1]gamereplays.org, drop.sc, and sc2rep.com.

Table 2: Visuospatial attention and motor skills.

Feature per-minute	Feature over-all	Description
Macro	Avg. macro	Macro actions
Micro	Avg. micro	Micro actions
Actions-per-minute (APM)	Avg. APM	Sum of macro and micro actions
Effective APM (EAPM)	Avg. EAPM	Effective actions
n/a	Redundancy	Ratio of ineffective actions
Hotkeys used	Total hotkeys used	Number of times hotkeys are used
Hotkeys set	Total hotkeys set	Total of hotkeys set
Hotkeys added	Total hotkeys added	Number of times new hotkeys are assigned
n/a	Different hotkeys	Total number of different hotkeys
n/a	Hotkey use	Ratio of hotkeys used per hotkeys set

We have four groups of per-minute and over-all data:

1. *Visuospatial attention and motor skills* (Table 2) encompass mainly the total number of effective actions. We assume that expert players will make faster decisions, and will perform fewer redundant actions than novice players; they will also use hotkeys more effectively. To decide whether an action is macro, micro, and/or effective, we used the rules given by the `sc2gears` program. Typically, an action is considered 'macro' if it costs minerals or gas, otherwise it is a 'micro' action. Effectiveness of an action is decided by rules derived from expert knowledge. An example is that an action that gets canceled within one second after being performed, is not effective.
2. *Economy* (Table 3) encompasses the delicate balance that a *Starcraft II* player must find between collecting resources and building an army. The features that we measure to assess a player's economy encompass bases, workers, and resources spent.
3. *Technology* (Table 4) encompasses a player's technological development; in *Starcraft II* a player must maintain an effective balance between gaining technological advancements and defending his position. Besides counting technologies, the technological features that we use also encompass a player's 'tier', which is a general assessment of his overall technological development.
4. *Strategy* (Table 5) encompasses playstyle, consisting of a balance between offensive and defensive play.

Classifier selection

To select a suitable classifier to determine a player's league from their gameplay behavior, we performed a pretest in which we compared the performance of four classifiers using the *Weka* environment (Witten and Hall 2011): SMO (Sequential Minimal Optimization, a Support Vector Machine method), J48 (an Open Source implementation of the

Table 3: Economy features.

Feature per-minute	Feature over-all	Description
n/a	Bases total	Number of bases
Workers	Workers total	Number of workers built
Resources	Resources total	Sum of minerals and gas spent
Minerals	Minerals total	Minerals spent
Gas	Gas total	Gas spent
n/a	Workers per collection	Ratio of number of workers and number of gas collection buildings
n/a	Minerals per worker	Ratio of minerals spent and workers built

Table 4: Technology features.

Feature per-minute	Feature over-all	Description
n/a	Upgrades	Total number of upgrades researched
n/a	Abilities	Total number of special abilities acquired
n/a	Tier	Level of technological advancement

C4.5 algorithm), IBk (k-Nearest Neighbor), and RandomForest (an ensemble learner). We found that SMO outperformed all the other classifiers in accuracy by a good margin (see Figure 1), in particular for the bronze, platinum, and grandmaster leagues (Avontuur 2012).

Note that classifying instances to the master and grandmaster leagues is a relatively easy task for all classifiers. We assume that this is because those contain the best players, who have a consistent, easily recognizable play style. Also note that all classifiers have a hard time placing players in the silver and gold leagues. A possible explanation is that the classifiers assign a wide range of behaviors to the bronze league, even behaviors normally associated with players that belong to the silver and gold leagues. This happens because even relatively strong players start their career in the bronze league; thus, the classifiers are trained to assign silver and gold league behaviors to the bronze league. As accuracy is based on correctly classified instances, for the bronze league it remains high even if silver and gold league players get misclassified to the bronze league; however, this explains the low performance for the silver and gold leagues.

Results

Based on the results of the pretest, we used the SMO classifier to build a player model, that predicts which league a player belongs to based on their behavior as described by our feature set (Tables 2, 3, 4, and 5). We tested the performance of the player model using 5-fold cross validation. To examine which features contribute the most to league classification, we used InfoGain. Finally, we investigated how long a player must be observed before a correct prediction of their league can be made. A detailed description of all results is given by Avontuur (2012).

Table 5: Strategy features.

Feature per-minute	Feature over-all	Description
Supplies	Supplies total	Supplies used
n/a	Supplies gained total	Supplies gained by constructing supply buildings
n/a	Different units	Number of different unit types built
n/a	Fighting units	Number of units built that can fight
n/a	Defensive structures	Number of structures built that can deal damage

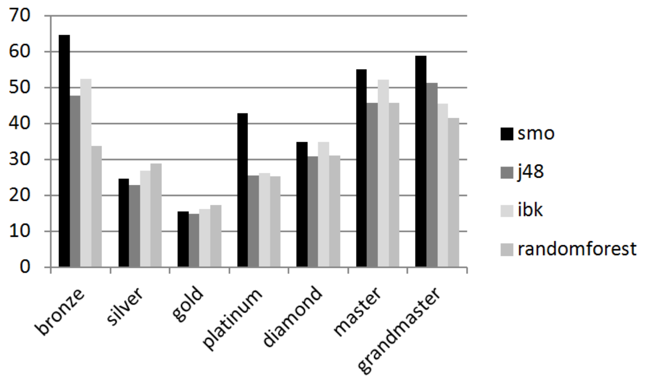

Figure 1: Comparison of accuracy of four classifiers on the dataset.

Player model performance

Table 6 gives an overview of the performance of the SMO-built player model expressed as the accuracy on each of the classes after 5-fold cross validation, including the average of the accuracies, the weighted average of the accuracies (i.e., with the contribution of each class in proportion to the number of instances in the corresponding league), and the majority-class baseline accuracy (which, in this case, is the percentage of instances belonging to the master league). The standard deviation is given between parenthesis.

The player model outperforms the frequency baseline by a large margin. A paired t-test shows that the accuracy of the player model with $t(4) = 14.35$, $p < .001$, and the weighted accuracy with $t(4) = 32.10$, $p < .001$, are significantly higher than the baseline, with an effect size $r = .99$ for both.

As the leagues are ordinal, a misclassification of an instance in a neighboring class can be considered less of a problem than a misclassification in more distant classes. Since there is overlap of player quality on the borders of the classes, such misclassifications into neighboring classes are actually to be expected. The confusion table (Table 7) shows that on average 67.0% of misclassifications are assigning an instance to a neighboring class.

We can estimate the distance of the misclassification as follows. First, we multiply the number of misclassified instances by the distance of their misclassification (e.g., the

Table 6: Player model performance.

League	Accuracy	
bronze	69.6%	
silver	25.8%	
gold	10.6%	
platinum	40.2%	
diamond	42.9%	
master	63.3%	
grandmaster	62.1%	
average	44.9%	(2.7)
weighted average	47.3%	(2.2)
majority-class baseline	25.5%	(1.1)

Table 7: Confusion table for player model.

bron.	silv.	gold	plat.	diam.	mast.	gr.m.	
567	135	39	57	16	2	0	bron.
206	**202**	127	185	63	12	1	silv.
80	195	**109**	394	178	69	14	gold
60	99	130	**649**	391	262	23	plat.
16	28	77	319	**865**	609	102	diam.
9	17	24	196	471	**1615**	217	mast.
0	0	2	9	95	331	**713**	gr.m.

number of gold instances classified to gold gets multiplied by zero, the number of gold instances classified to silver or platinum gets multiplied by 1, the number of gold instances classified to bronze or diamond gets multiplied by 2, etc.). For each league, we add up these numbers, and divide the total by the number of instances in the class. This is the average distance of the misclassification, which is displayed in the second column of Table 8. The closer the number is to zero, the better the classifications are; since a value of 1 is a placement in a directly neighboring class, everything below 1 means that the classifications are quite accurate. The third column of Table 8 is calculated in a similar way, but only takes into account incorrectly-classified instances, i.e., it indicates the average distance of the misclassification. The closer this number is to 1, the more likely it is that misclassifications put an instance into a neighboring class.

We can see that on average, the distance of the misclassification is 1.55. This means that most instances were placed in a neighboring class. This is also true for each individual class, as the average distances are lower than 2 for all classes.

Contribution of features

InfoGain assigns a score to the features according to the information they provide in solving the classification problem. We applied InfoGain to all five training sets used in the 5-fold cross validation. The top 8 features were ranked the same for each of the five folds, namely, in order: (1) average micro (over-all), (2) average APM (over-all), (3) EAPM (over-all), (4) micro (per-minute), (5) EAPM (per-minute), (6) hotkeys used (per-minute), (7) APM (per-minute), and (8) total hotkeys used (over-all). Figure 2 plots the average weights of the features according to their rank. It shows a sharp drop in InfoGain after the eighth ranked feature (from 0.42 to 0.26).

Table 8: Average misclassification distance.

League	Avg. classification	Avg. distance of misclassifications
bronze	0.75	1.95
silver	1.34	1.75
gold	1.47	1.61
platinum	1.11	1.67
diamond	0.92	1.35
master	0.60	1.35
grandmaster	0.38	1.19
average	0.94	1.55

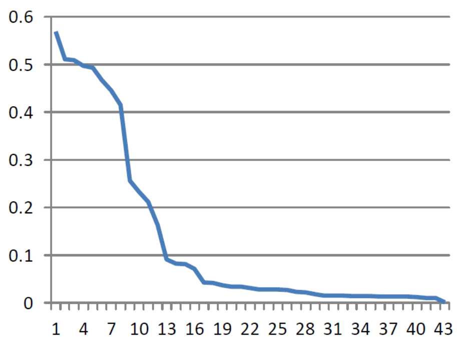

Figure 2: Average weight of features according to InfoGain.

If we remove the 20 lowest-ranked features from our training set and build the model on the remainder, we find that the weighted accuracy of the resulting model drops from 47.3% (2.2) to 43.8% (1.1). A paired t-test shows that this is a significant difference ($t(4) = 2.93, p < .05, r = 0.88$). However, the time needed to build the model is also reduced by a factor of almost 50 (from 2400 to 49 seconds).

Time dependence

As about half the features of the player model are calculated as averages over gameplay time, we may expect that the performance of the player model improves the longer the game is played. To test this, we calculated the model's weighted accuracy using a 5-fold cross validation over ten different sets: one set with all the gameplay features up to and including the first recorded minute (i.e., up to the first 2.5 minutes of gameplay), one up to and including the second gameplay minute, etc. The results are shown in Figure 3, where the upper line indicates the weighted accuracy of the player model, and the lower line indicates the weighted accuracy of the baseline, progressing through time.

It is clear from the figure that the accuracy of the player model does not change much over time. To determine whether the observed small increase in accuracy is significant, we performed a series of tests. Applying an ANOVA shows that time has no significant effect on the accuracy of the player model ($F(8) = .082, p > .05$). We may therefore conclude that the player model's classification is as accurate as it can be after the first recorded minute of gameplay (2.5 minutes into the game). This is, in general, before any substantial confrontations with the enemy have taken place.

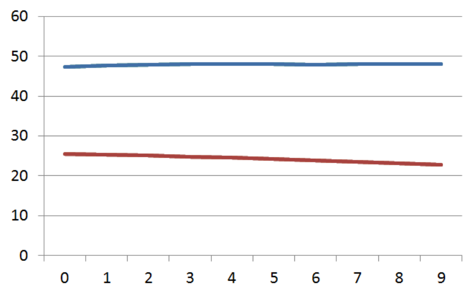

Figure 3: Weighted accuracy (percentage) of the player model for different periods.

Discussion

In this section we make some observations on the player model's performance, the model's features, and the possibilities to use the player model to create a game-playing AI.

Player model performance

Our results show that our player model is able to classify players according to their league significantly better than the baseline, and that misclassifications on average place a player in a neighboring class. We argue that most misclassifications can be attributed to the nature of the data set. The classes are the leagues to which players are assigned by Blizzard's ladder system. This system incorporates measures to ensure that players whose skills are on the border of two leagues, do not switch leagues too often. For instance, if a player of the silver league defeats the lower-performing players of the gold league, but not those of average performance, he will not be promoted to the gold league. Also, a silver-league player who consistently defeats gold-league players but does not play often, will not be promoted quickly. Therefore, we must assume that the borders of two neighboring leagues overlap quite a bit.

To build a player model of higher accuracy than we achieved, we would need to have access to more specific rankings of a player, e.g., like an ELO-rating that assigns a player an individual rank in comparison with all his fellow players. Blizzard actually calculates such a rank, which is one of the elements used to determine when they allow a player to switch leagues. However, at present these 'hidden rankings' are invisible to the outside world. Our data set is not sufficiently large to calculate a reliable ELO-rating of our own.

Features

We used InfoGain to determine which features of the model contribute most to the classification of players. The three highest ranked features are (i) average micro, (ii) average APM, and (iii) average EAPM. All three features describe player behavior over the game until the moment of measurement. This indicates that the game history is an important factor in determining the skill of a player.

The top-eight features all measure motor skills. They show that a gamer must have excellent control over his mouse and keyboard to issue a large number of commands and use a large number of hotkeys in a short period of time. Moreover, the two highly-ranked features which involve hotkeys show that strong players need excellent visuospatial attention. These observations coincide with the conclusions drawn by previous researchers on the strength of video game players (Green and Bavelier 2006; 2007; Dye, Green, and Bavelier 2009).

AI implementation

Now we have acquired some understanding of how to recognize the skill levels of human players in *Starcraft II*, we discuss two potential venues to apply this knowledge: (1) using it to create stronger AI by imitating the behavior of strong human players, and (2) creating an adaptive AI that scales to the observed strength of the opposing human player.

Our findings do not provide much help in following the first venue: learning from the model to create a more effective AI. InfoGain ranked micro-management of units through a high number of actions (and effective actions in particular) as the most important features of the player model. That only tells us that a strong AI should have effective micro-management. It does not, however, indicate what 'effective micro-management' entails. Moreover, while strong human players distinguish themselves from the weaker ones by the speed by which they can give commands, such speed is not an issue for a computer, and thus not for an AI. Finally, some of the high-ranked features, such as the use of hotkeys, are only meaningful to describe human players, not AI players.

However, our findings do provide help in following the second venue: the creation of an AI that adapts its difficulty level to the observed strength of the human player. Since the player model offers us the ability to recognize the human opponent's strength with high accuracy already early in a game, there is sufficient game-time left to make simple changes to, for instance, the AI's economy or tactics. Downgrading the effectiveness of an AI is not hard, by squandering resources or building less effective units.

Conclusion

We used the SMO classification algorithm to build a player model that recognizes the league that *Starcraft II* players are assigned to. The model achieves a weighted accuracy of 47.3%, which is significantly and substantially over the majority-class baseline. Moreover, 67.0% of misclassifications assign a player to a neighboring league. Taking into consideration how players are assigned to and switch between leagues, players being positioned one league lower or higher than their skill level is actually common, and thus such misclassifications are only to be expected. We conclude that we have been able to create a player model that recognizes a player's league with high accuracy.

We found that the most distinguishing features of our player model are based on visuospatial and motor skills of players. It is particularly effective at recognizing novices and high-level players. Our findings show that we can detect a player's league already in the first minutes of a game, which indicates that an AI can use this information to adapt its difficulty to the human player's observed skill level.

References

Avontuur, T. 2012. *Modeling Player Skill in Starcraft II*. HAIT Master Thesis series 12-004. Tilburg, The Netherlands: Tilburg University.

Bakkes, S.; Spronck, P.; and van Lankveld, G. 2012. Player behavioral modelling for video games. *Entertainment Computing* 3(3):71–79.

Campitelli, G., and Gobet, F. 2004. Adaptive expert decision making: Skilled chess players search more and deeper. *Journal of the International Computer Games Association* 27(4):209–216.

Campitelli, G.; Gobet, F.; Williams, G.; and Parker, A. 2007. Integration of perceptual input and visual imagery in chess players: Evidence from eye movements. *Swiss Journal of Psychology* 66:201–213.

Canossa, A. 2009. *Play-Persona: Modeling Player Behaviour in Computer Games*. Ph.D. thesis. Copenhagen: Danmarks Design Skole.

Carmel, D., and Markovitch, S. 1993. Learning models of opponent's strategy in game playing. In *Proceedings of AAAI Fall Symposium on Games: Planning and Learning*, 140–147.

Chaddock, L.; Neider, M.; Voss, M.; Gaspar, J.; and Kramer, A. 2011. Do athletes excel at everyday tasks? *Medicine & Science in Sports & Exercise* 43(10):1920–1926.

Chase, W., and Simon, H. 1973. Perception in chess. *Cognitive Psychology*.

Drachen, A.; Canossa, A.; and Yannakakis, G. 2009. Player modeling using self-organization in tomb raider: Underworld. In *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, 1–8.

Dye, M.; Green, C.; and Bavelier, D. 2009. The development of attention skills in action video game players. *Neuropsychologia* 47:1780–1789.

Gobet, F., and Simon, H. 1996. The roles of recognition processes and look-ahead search in time-constrained expert problem solving: Evidence from grand-master-level chess. *Psychological Science* 7(1):52–55.

Gobet, F., and Simon, H. 1998. Expert chess memory: Revisiting the chunking hypothesis. *Memory* 6(3):225–255.

Green, C., and Bavelier, D. 2006. Enumeration versus multiple object tracking: The case of action video game players. *Journal of experimental psychology. Human perception and performance* 32(6):1465–1478.

Green, C., and Bavelier, D. 2007. Action-video-game experience alters the spatial resolution of vision. *Psychological Science* 18(1):88–94.

Iida, H.; Uiterwijk, J.; van den Herik, H.; and Herschberg, I. 1993. Potential applications of opponent-model search. part 1: the domain of applicability. *ICCA Journal* 16(4):201–208.

Kocsis, L.; Uiterwijk, J.; Postma, E.; and van den Herik, H. 2002. The neural movemap heuristic in chess. In *Computers and Games*, 154–170.

Reingold, E.; Charness, N.; Pomplun, M.; and Stampe, D. 2001. Visual span in expert chess players: Evidence from eye movements. *Psychological Science* 12(48).

Schadd, F.; Bakkes, S.; and Spronck, P. 2007. Opponent modeling in real-time strategy games. In Roccetti, M., ed., *8th International Conference on Intelligent Games and Simulation*, 61–68.

Schriver, A.; Morrow, D.; Wickens, C.; and Talleur, D. 2008. Expertise differences in attentional strategies related to pilot decision making. *Human Factors: The Journal of the Human Factors and Ergonomics Society* 50:864.

Tekofsky, S.; Spronck, P.; Plaat, A.; van den Herik, H.; and Broersen, J. 2013. Psyops: Personality assessment through gaming behavior. In *Proceedings of the FDG 2013*.

van den Herik, H.; Donkers, H.; and Spronck, P. 2005. Opponent modelling and commercial games. In Kendall, G., and Lucas, S., eds., *Proceedings of the IEEE 2005 Symposium on Computational Intelligence and Games (CIG'05)*, 15–25.

van der Werf, E.; Uiterwijk, J.; Postma, E.; and van den Herik, H. 2002. Local move prediction in go. In *Computers and Games*, 393–412. Springer.

Van Lankveld, G.; Spronck, P.; van den Herik, H.; and Arntz, A. 2011. Games as personality profiling tools. In *2011 IEEE Conference on Computational Intelligence in Games (CIG'11)*.

Weber, B., and Mateas, M. 2009. A data mining approach to strategy prediction. In *IEEE Symposium on Computational Intelligence in Games (CIG 2009)*, 140–147. IEEE Press.

Witten, F., and Hall, M. 2011. *Data Mining: Practical Machine Learning Tools and Techniques (Third Edition)*. New York: Morgan Kaufmann.

Yee, N. 2006. Motivations for play in online games. *CyberPsychology & Behavior* 9(6):772–775.