Investigation of the Effect of "Fog of War" in the Prediction of StarCraft Strategy Using Machine Learning

HOCHUL CHO, HYUNSOO PARK, CHANG-YEUN KIM, and KYUNG-JOONG KIM,
Sejong University

StarCraft is a well-known real-time strategy game developed by Blizzard Entertainment in 1998. One of the characteristics of this game is "fog of war," which refers to the fact that players cannot see their opponents' regions but only their own unit. This characteristic of the game means that the information required in order to predicting the opponent's strategy is only available through "scouting." Although the "fog of war" is one of the most important features of the game, it has not been deeply understood in the design of artificial intelligence. In this work, we propose to investigate the effect of the "fog of war" in the prediction of opponent's strategy using machine learning for human players and artificial intelligence (AI) bots. To realize this analysis, we develop a customized replay analyzer that exports the internal game events with/without the fog of war. In the experimental results, we collect replays from various sources: human vs. human, human vs. AI bots, and AI bots vs. AI bots. This systematic analysis with "fog of war" reveals the predictability of the machine-learning algorithms on different conditions and the directions for designing new artificial intelligence for the game.

CCS Concepts: • **Computing methodologies** → **Machine learning approaches**; *Artificial intelligence*; • **Information systems** → *Decision support systems*; *Data mining*

Additional Key Words and Phrases: Real-time strategy game, starcraft, machine learning, strategy prediction, fog of war, AI bots

ACM Reference Format:
Hochul Cho, Hyunsoo Park, Chang-Yeun Kim, and Kyung-Joong Kim. 2017. Investigation of the effect of "fog of war" in the prediction of starcraft strategy using machine learning. Comput. Entertain. 14, 1, Article 2 (January 2017), 16 pages.
DOI: http://dx.doi.org/10.1145/2735384

1. INTRODUCTION

In most real-time strategy (RTS) games, winning or losing is determined by eliminating all of the opponent's units and buildings. To achieve this, a player should gather resources, build buildings and units, and attack the opponent's territory. However, in-game behavior is complicated by so-called "'fog-of-war," which renders the opponent's territory invisible. To overcome this uncertainty, a player must send "scouting" units into enemy territory, providing limited visibility of the territory around the opponent. The level of the player's skills is very important in determining how successful scouting

Ruck Thawonmas, Guest Editor
This research was supported by the Basic Science Research Program and the Original Technology Research Program for Brain Science through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0012876 and 2010-0018948).
Authors' addresses: H. Cho, H. Park, C.-Y. Kim, and K.-J. Kim, Department of Computer Science and Engineering, Dasan Gwan 124B, Sejong University, 209, Neungdong-ro, Gwangjin-gu, Seoul, Republic of Korea 05006; emails: chc2212@gmail.com, hspark@sju.ac.kr, magnate0@naver.com, kimkj@sejong.ac.kr (corresponding author).

DOI: http://dx.doi.org/10.1145/2735384

can be in predicting the opponent's strategy from this partial information. Prediction is an important aspect of human behavior, and in a game situation, winning or losing is strongly dependent on the ability to predict an opponent's actions.

StarCraft is a well-known RTS game developed by Blizzard Entertainment in 1998 that exhibits fog of war. In the game, each player continuously scouts the opponent's territory for information about the opponent's buildings and units and uses this (incomplete) information to make decisions on his or her strategy. This aspect of the game is critical to the win rate. However, despite the importance of the fog of war, there are few works on the prediction of strategy with the feature. For example, Weber and Mateas [2009] simulate the fog of war with random "missing" of observed units. Park et al. [2012] propose to use a scouting unit developed for game artificial intelligence (AI)bots to collect realistic game logs against human players. The shortcoming of the approach is that the technique is not applicable to the replay files not played by the AI bots. In this article, we develop a customized replay analyzer to extract information from any kinds of replay files with/without fog of war. With the tool, it is attempted to investigate the effect of "fog of war" to predict the strategy using machine learning.

Predicting an opponent's strategy using this incomplete information represents a challenge for AI. Weber and Mateas [2009] proposed using professional player's replay files as a source of data mining. It is possible to obtain text-based information on player's actions in the game using specialized programs called replay browsers [Lord Martin's Replay Browser]. They reported that machine learning can be useful in predicting the strategy of an opponent in the early stage of the game. In their analysis, they concluded that visibility of information is critical. First, they assumed that there was no fog of war and that all the information on the opponent could be used to predict strategy. In the second refinement, they added random noise and missing data to simulate the fog of war. This simulation was not realistic, however, because of the limitations of the replay browser used. Several attempts have been made to use replay files to build strategy-prediction models [Kim and Cho 2011; Hsieh and Sun 2008]. However, they all suffer from the same unrealistic settings.

Our research is motivated by the StarCraft AI competition [Buro and Churchill 2012], conferences for computer game AI (IEEE CIG[1] and AAAI AIIDE[2]), and the open competition for StarCraft AI [Kim and Cho 2012; AIIDE StarCraft AI Competition]. It is still quite challenging to develop AI for the game because it should handle a number of units and buildings while considering resource management and high-level tactics as well as the fog-of-war problem. Unlike humans, the bots usually have limited ability to control "scouting unit," predict "opponent's strategy," and make decisions on build order change. Because the executables and source codes of the AI bots have been opened to the public, it is possible to test the state-of-the-art AI bots in this investigation.

Although the fog of war is forced in the game of StarCraft, we can control the feature in the analysis of replay files. From thousands of replay files from the Internet for various types of games (human vs. human, human vs. bots, and bots vs. bots), it is possible to test the prediction ability of the machine-learning algorithms for different conditions. It is expected that the prediction ability should be maximized when we turn off the fog of war. Although it is not realistic, this performance could be used as an upper bound of the prediction. With fog of war, the prediction performance might be decreased but the degradation should differ depending on the types of machine-learning algorithms and players. From the analysis, it gives invaluable information on the predictability of machine-learning algorithms with uncertainty of the game.

[1]IEEE Conference on Computational Intelligence and Games.
[2]AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment.

Fig. 1. An in-game screen shot showing bot playing StarCraft.

2. BACKGROUND

2.1. StarCraft AI Competitions

As with other commercially available programs, the most difficult aspect of making an AI bot for StarCraft is that the source code is not available. Using the open-source Brood War Application Programming Interface (BWAPI), built in 2009 specifically for StarCraft, developers can acquire information on units, buildings, and regions. Furthermore, they can also control various units in the game. Following the release of BWAPI, StarCraft AI contests have been held, and much related research has been carried out. Figure 1 shows in-game screen AI bot playing StarCraft.

The first StarCraft AI competition was part of the AAAI conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2010). More than 26 teams registered, and the final winner of the complete game was Overmind from UC Berkeley. The 2011 IEEE Conference on Computational Intelligence and Games (CIG 2011) included a StarCraft AI competition. In that event, 10 teams registered, and the winner was Skynet. Four bots made it to the final round of CIG 2011: Skynet, UAB, Xelnaga, and BotQ [Synnaeve and Besiere 2011a; IEEE CIG StarCraft AI Competition 2011]. Each bot played 30 games, and Xelnaga won 11 games.

2.2. StarCraft Scouting in Bots

In StarCraft, novice players build their buildings and produce units without much attention to opponent’s plays. Experienced players start the game by sending their construction units outside of their territory for scouting. The purpose of the scouting is to see the buildings and units in the opponent’s area. If the scouting unit successfully enters into the area, then it removes the fog of war around the unit if the unit is alive. Although the removal is limited to the near area around the unit, it is very useful to infer opponent’s build orders. Because the scouting unit leaks important information, the opponent tries to kill the scouting unit and/or delay their construction of critical units hiding their build orders.

Table I summarizes the state of the art of the scouting for the StarCraft AI bots. It shows that only small number of participants implement the scouting. Most of them use the scouting to disturb opponents rather than observing opponents. SPAR is one of bots perform scouting behavior and it is similar to the human’s one. However, there is no article on their approaches.

Table I. Comparison of StarCraft AI Bots' Scouting Capabilities

AI Bots	Race	Scouting	Comments
Aiur	Protoss	X	
BTHAI	Zerg	X	
EvoBot	Terran	X	
Nova	Terran	O	Disturbance
Skynet	Protoss	O	Disturbance
BotQ	Protoss	X	
LSAI	Zerg	O	Overload
UalbertaBot [Churchill and Buro 2012]	Protoss	O	Disturbance
Beast Jelly	Protoss	X	
Bigbrother	Zerg	X	
Cromulent	Terran	X	
EISBot [Weber et al. 2011]	Protoss	X	
ItalyUndermind	Zerg	O	Disturbance
Quorum	Terran	X	
SPAR	Protoss	O	Observation
Undermind	Terran	X	

2.3. Prediction of Opponent Strategy Using Replays

Hsieh and Sun [2008] collected thousands of StarCraft game replays from GosuGamers game community. They used case-based reasoning to learn and predict individual player's strategies. In their work, cases in Case-Based Reasoning (CBR) contain constructed buildings, military units, money, and other resources. They reported that predictive accuracy increased when more replays were inputted into their decision system. Weber and Mateas [2009] collected 5,493 StarCraft game replays from GosuGamers.net and TeamLiquid.net, two popular StarCraft portals. They encoded replays as feature vectors and labeled them using rules based on analysis of expert plays. They formulated a strategy prediction problem as a multi-class classification problem and applied decision tree, K-Nearest Neighbor (KNN), Non-Nested Generalized Examplars NNGE, and LogitBoost. They reported that the machine-learning algorithms could predict the strategies in the early stage of the game. In particular, NNGE and KNN performed well in the initial stage of the game, but degraded in the later stages of the game.

Weber and Ontanon [2010] et al. used StarCraft replays in order to build a case library for D2, a real-time case-based planning system designed to play RTS games. They defined goal ontology for StarCraft and developed a Rule Set for recognizing when goals are being pursued. The goal ontology was formulated based on analysis of professional StarCraft game play.

Kim et al. [2010] tried to analyze build orders and the relations among them from game replays. For example, they figured out how many times a build order A won against build order B. They used similarity measure on unit production, building construction, and upgrade order to categorize build orders. They united build-orders as a tree to choose appropriate one based on enemy's state and winning ratio. Besides, there was various research based on modeling methods for prediction, which are a hierarchically structured model [Kabanza et al. 2010], a Bayesian model [Synnaeve and Besiere 2011b], and an Hidden Markov Model (HMM) [Dereszynski et al. 2011].

3. LEARNING STRATEGY PREDICTION WITH FOG OF WAR

In the proposed method (Figure 2), the replays are coming from multiple sources. In Weber and Mateas [2009], the replays are from game portals and they're all human vs. human. In Park et al. [2012], the replays are created from the private Battle Net server

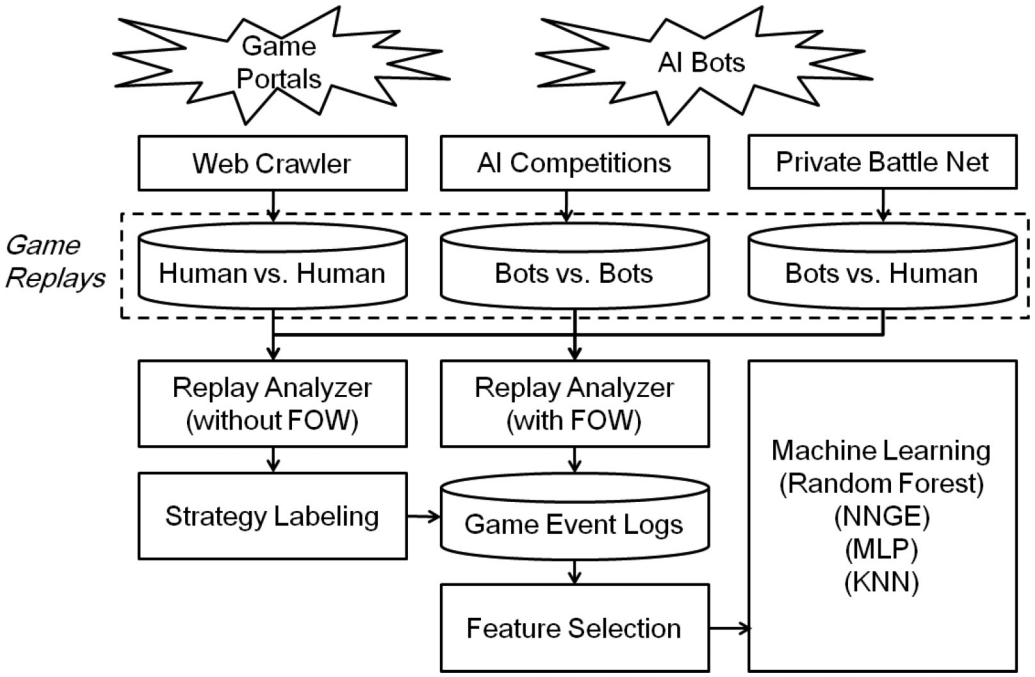


Fig. 2. An overview of the proposed method (FOW = Fog of War).

using AI bots against human players. In this work, we consider the two sources and the bots vs. bots replays from the replay repository of the AI competitions. Although the history of the AI competition is not too long (just 3 years so far), it runs full round-robin style tournaments and produces a lot of replays among bots.

- Replay Analyzer (without FOW): This is a technique that the previous works on replay mining usually adopted. In this approach, Lord Martin Replay Browser and BW Chart have been used to extract gaming logs from them. Because the tools do not support the extraction of events considering the fog of war. It just extracts all the events that are not visible to the player during the game. So, the information is not realistic to reflect the game situation.
- Replay Analyzer (with FOW): In this work, we develop a customized replay analyzer using BWAPI (open software development kit (SDK) for StarCraft AI programming). Unlike the tools, the analyzer reflects the fog of war and outputs only gaming events visible to each player's vision. The analyzer loads the replay file and plays the game in highest speed (about 8 times faster than normal playing). During the game, the analyzer participates in the game as an observer role and records only visible gaming events to each player. In this way, the analyzer extracts gaming logs with fog of war with small amount of time.
- Strategy Labeling: It is necessary to label "strategy" of replay to be used as training samples for machine learning. If human experts are available, then he or she can review the games and label them manually. However, it is not feasible because the number of replays to be analyzed is large (more than hundreds and sometimes thousands). It is recommended to define a set of rules (Rule Set) that classify the replays into one of known "strategy" based on full information of the game (without fog of war).

(a) Screen shots

155 Protoss_Probe 132
155 Protoss_Nexus 130

155 Protoss_Probe 132
155 Protoss_Nexus 130
155 Protoss_Probe 133
155 Protoss_Probe 134
155 Protoss_Probe 135
155 Protoss_Probe 136
155 Protoss_Probe 137
155 Protoss_Probe 138
155 Protoss_Probe 139
155 Protoss_Probe 140
155 Protoss_Probe 141
155 Protoss_Pylon 131
155 Protoss_Probe 142

(b) Game logs

Fig. 3. Comparison of screen shots and game logs extracted from replays with/without fog of war (on the left, the scouting unit sees only one probe and the nexus and others are invisible because of the fog. However, all the units and buildings are stored without fog of war.)

- Feature Selection: In this step, the gaming events are converted into a feature vector. Each feature is corresponding to units or buildings. It stores the first time that each object is created. The challenge of the prediction is to predict the "strategy" in the early stage of the game where the construction and production is under development. To train prediction models suitable for the early stage prediction (at time t), the feature stores information only played before the time t.
- Machine Learning: In this step, the feature vectors and labels are used to train prediction models. Although there are a lot of learning algorithms, in this work, we propose to use Random Forest, NNGE, MLP (Multi-Layer Perceptron), KNN. From previous works, they show promising results to predict RTS strategy. It repeatedly trains models suitable for prediction at time t.

3.1. Replay Analysis with Fog of War

Figure 3 shows the difference of the game screen with/without fog of war. In the fog of war, the territory of opponent is hidden and the limited area around the player's unit is visible. Without fog of war, the units and buildings of opponent are fully visible. The fog of war makes the strategic decision is more important than other factors in the game. Because my opponent has limited visibility to my actions, it is possible to design "strategy." If all actions are visible to each other, then the power of strategy

Raw Data	Features	
	Structure/Units	Time
...		
122 Protoss_Probe 138		
123 Protoss_Probe 138		
123 Protoss_Nexus 139	Protoss_Nexus_1	86
124 Protoss_Probe 138	Protoss_Probe_1	87
124 Protoss_Probe 144	Protoss_Probe_2	87
124 Protoss_Gateway 147	Protoss_Probe_3	88
124 Protoss_Probe 143	Protoss_Assimilator_1	91
124 Protoss_Probe 141	Protoss_Pylon_1	94
124 Protoss_Probe 148	Protoss_Gateway_1	110
124 Protoss_Probe 149	Protoss_Pylon_2	140
124 Protoss_Probe 145	Terran_Factory_1	169
124 Protoss_Probe 146	Protoss_Zealot_1	184
124 Protoss_Probe 142	Protoss_Zealot_2	203
124 Protoss_Nexus 139	Protoss_Pylon_3	208
125 Protoss_Probe 144		
125 Protoss_Gateway 147		
...		

(b)

Fig. 4. Examples of features extracted from raw gaming logs (The game logs show the events at 123∼125 game time.) The features summarize the logs.

might be minimized. Our replay analyzer extracts information in two different modes (with/without fog of war). If the fog of war is considered as in a real game, then the event logs contains only information visible to each player (the game logs can be stored for each player). Although the logs extracted without fog of war (all actions of opponents are stored) usually are useful to build "accurate" prediction models, they're not successful to handle the actual game with fog of war.

In the logs, the first column represents the time stamp of the events. The second column shows the type of unit or buildings. The last column is the unique ID (Identification) number for the object. Each row represents the data for each object. For each time stamp, it contains a log of event logs visible to the player. Because the number of unique events is very large for a single game, it is necessary to delete unnecessary information (for example, mouse click event) and merge successive actions in to single one.

3.2. Feature Creation

Using the BWAPI StarCraft AI programming SDK, it is possible to extract the types of observed objects together with a unique ID. Figure 4 shows an example of recorded raw data set from the replay analyzer, where the opponent race was Protoss. At 122s from the beginning of the game, the recon unit observed one Protoss probe unit (ID = 138). At 123s, a Protoss probe (ID = 138) and a Nexus (ID = 139) were in view. At 124s, 11 units or buildings were visible.

The raw data contain all the information observed during scouting. However, it is not easy to make decisions about the opponent's strategy from the raw data. It is necessary to fuse information from the multiple scenes into one summarized data set. Figure 4 shows the time at which the units were observed. Here, Protoss_Nexus_1 shows the time at which the first command center was observed (at 86s). This does not

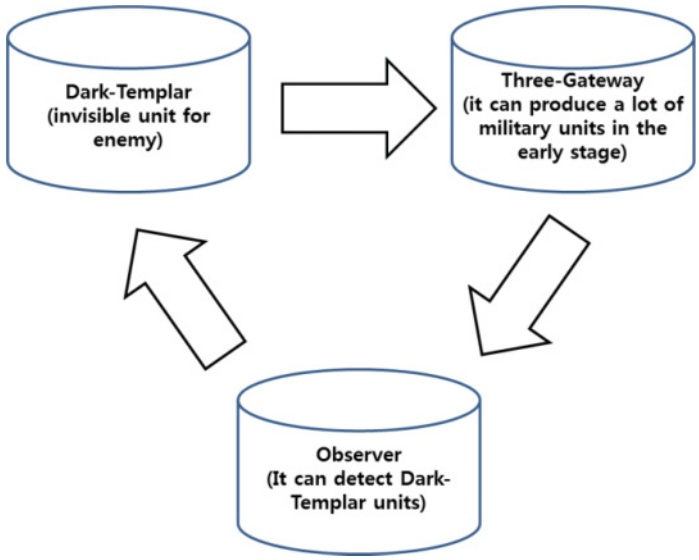


Fig. 5. The relative advantages of the three commonly used strategies in StarCraft. The arrow points from the advantageous strategy to the disadvantageous one.

mean that the Nexus was constructed at that time. Protoss_Probe_1, Protoss_Probe_2, and Protoss_Probe_3 show the observed times for Probe_1 (the first Probe observed), Probe_2 and Probe_3, respectively. From the data, it is not easy to figure out the mass of the units or buildings because the system only records the times for the first two or three units or buildings.

3.3. Prediction of an Opponent's Strategy

The final stage is to a make decision about the opponent's strategy using the fused data. If expert knowledge is available, then it can be stated as simple rules. For example, if the number of Zealots (the attack unit of the Protoss race) is larger than 15, the strategy is named as "three gate." It is necessary to add multiple rules to determine the type of an opponent's strategy. Although this is simple, it is not trivial to design the rules. The first problem is that this kind of expert knowledge is not easily defined. Second, the information from the scouting unit is incomplete (i.e., there may be unobserved or hidden information). Additionally, some units or buildings may be killed or destroyed after they were observed. Furthermore, the scouting period may be too short to retrieve sufficient information.

The solution to the problems is to apply machine learning using collected data. Because there are three race types in StarCraft, there are six possible types of game. However, we focus on Protoss vs. Protoss because the strategic prediction is so important in the competition. We categorized the opponent strategy into three categories that gamers usually refer as "Three Gateway," "Observer," and "Dark Templar." These strategies are well-known standard build orders among gamers, and each has its advantages and disadvantages, similarly to the rock-paper-scissors game. Figure 5 shows the relative advantages of the three strategies. Although these three build orders are important in determining the outcome of a game, predicting an opponent's strategy is not easy because these build orders may not easily be determined due to incomplete scouting data.

```
@relation ''
@attribute Protoss_Cyber1 real
@attribute Protoss_Robo1 real
@attribute Protoss_Adun real
@attribute Protoss_Observertory real
@attribute Protoss_Dragon_num real
@attribute Protoss_Zealot_num real
@attribute Protoss_Gateway1_time real
@attribute Protoss_Gateway2_time real
@attribute Protoss_Gateway3_time real
@attribute Protoss_Nexus1_time real
@attribute Protoss_Nexus2_time real
@attribute Protoss_Templar_archive_time real
@attribute Protoss_Probe_num real
@attribute Protoss_Pylon_num real
@attribute Protoss_gas_time real
@attribute class {3gate, observer,dark,nexus}
@data
151,226,0,0,5,2,76,248,0,0,0,0,26,4,99,3gate
151,226,0,0,5,2,76,248,0,0,0,0,26,4,99,nexus
156,0,204,0,2,3,85,264,0,0,0,251,23,3,106,dark
126,283,177,0,1,1,80,0,0,263,0,218,25,4,92,dark
156,0,0,0,4,2,77,295,0,258,0,0,25,4,124,3gate
286,0,0,0,0,12,96,99,218,0,0,0,19,7,281,3gate
160,292,0,0,4,3,78,255,0,0,0,0,26,5,140,observer
...
```

Fig. 6. An example of training data for a Protoss opponent (zero means that the unit/building was not observed).

Machine learning was carried out with and without fog of war. Data from the replay files were extracted and converted into a summarized form using the preprocessing (merging) algorithms, as shown in Figure 6. Then, a set of machine-learning algorithms was applied to the summarized data to build prediction models using the WEKA (Waikato Environment for Knowledge Analysis) which is open-source machine learning library [Witten et al. 2011]. The algorithms used were decision tree (Random Forest [Breiman 2001]), lazy learning (1-NN [Aha et al. 1991] and NNGE [Brent 1995]), and neural networks (MLP). Leave-one-out-cross validation (LOOCV) was used in all of our experiments. For each classifier, the default parameters in WEKA were used. We compare the results of machine learning and the Rule Set (designed by human experts assuming ideal observation of opponents), which initially had 33.3% prediction accuracy because there are three strategy types.

4. EXPERIMENTAL RESULTS AND ANALYSIS

4.1. Data Collection

We collected a lot of replays from different sources (Table II).

- Human vs. human: Since 1998, numerous StarCraft games have been played on BattleNet, and each player could upload replays on Internet sites individually. StarCraft is very popular in Korea, where there is even a StarCraft pro league; as a result, many replays are uploaded and are available. The Korean web site Ygosu.com maintains an archive of replays dating back to 2001 that includes approximately 78,000 replays. We selected high-ranking players and carried out data mining on Protoss vs. Protoss replays.

Table II. The Number of Data Samples and Labels (It Is Possible to Extract two Samples from One Replay File by Interchanging Their Roles (Player and Opponent))

(a) The Number of Data Samples

	Replays	Actual Samples in Data Mining	Resource
Human vs. human	369	738	Web Site (Ygosu.com)
Bots vs. bots	178	356	AIIDE 2010, 2011 and IEEE CIG 2011
Bots vs. human	49	49	Plays in private BattelNet server (Xelnaga Bot)

(b) The ratio (%) of Strategies According to the Types of Opponent (in the Case of Xelnaga vs. Human the Target Is Human)

	Three-gateway	Observer	Dark Templar
Human vs. human	37.5	48.2	14.3
Bots vs. bots	63.3	10.4	26.3
Bots vs. human	18.6	76.7	4.7

- Bots vs. bots: The StarCraft AI competition AIIDE started in 2010, and IEEE CIG started in 2011, providing three AIIDE and two IEEE CIG tournaments for which replay data are available. In 2012, both competitions became automated, so many games were held during that year. We used replays from the AIIDE 2011 and 2012 and IEEE CIG 2012 competitions.
- Bots vs. human: The most difficult problem for the StarCraft data mining is to collect data. Because we need data to train our scouting-based bots, the replay files accessible on the internet are not useful. Although we consider generating the data by playing games between our bots and other bots, there are not enough bots available. Our choice is to collect the data from the games between our bots and human players. We played 105 games against human players.

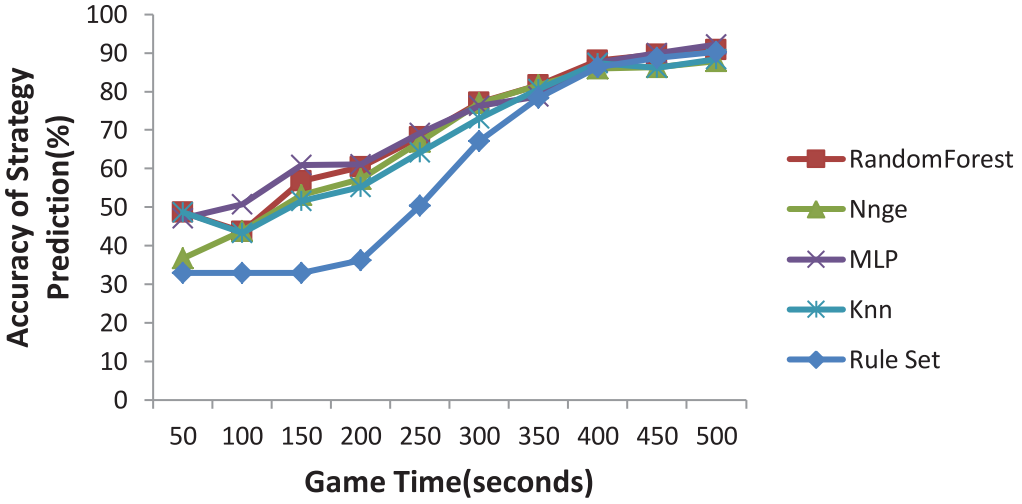
In human vs. human games, Observer is chosen most often; this is because Observer is stronger than Dark Templar and can scout faster than the other build orders can. The highest ratio is 48.2%, and this defines the ZeroR (rule algorithm), which simply classifies samples into the most frequent classes. If a result of machine learning is below ZeroR (i.e., 48.2%), then the machine learning was not useful.

In bots vs. bots, Three Gateway was the most frequently chosen strategy, and Observer was markedly lower, in contrast to the case in human vs. human games. There are two reasons for this. First, the bots did not possess the skills necessary for scouting the opponent's base, and, therefore, Observer was less effective. Second, in longer (and hence more complicated) games, because the game is more challenging, bots that favor strategies that finish the game early are more likely to win. Three Gateway fits this purpose.

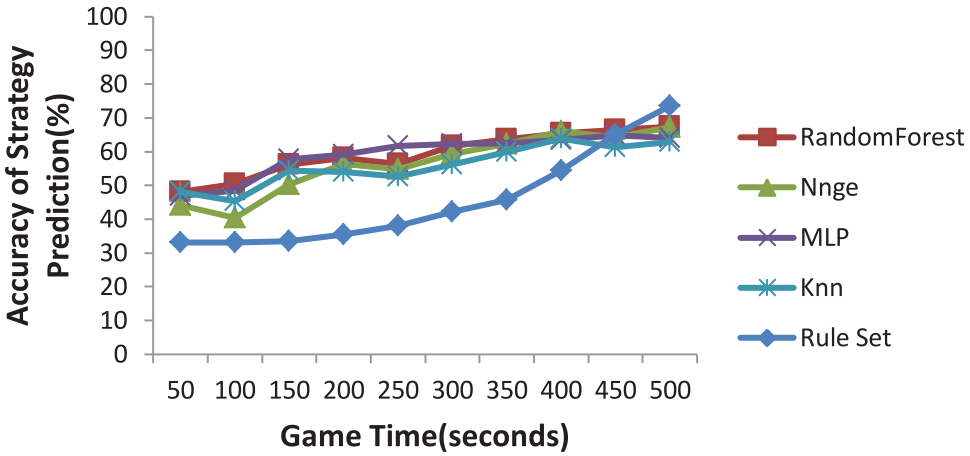
In the Xelnaga vs. human games, Observer is the most commonly chosen strategy for the human player. The reason is that the build order of Xelnaga is always Dark Templar, and we can see that if the human player is able to see the build order of Xelnaga by scouting, the human player will change the build order to Observer.

4.2. Strategy Prediction Results

4.2.1. Human Vs. Human. Figure 7 shows the accuracy of strategy prediction in human vs. human games. With no fog-of-war, after about 100s, the prediction accuracy starts to increase, until approximately 350s, where the accuracy saturates at around 90%. This shows that without fog of war, machine learning can effectively predict the opponent's strategy, which is in agreement with Weber's experiments [Weber and Mateas 2009]. However, prediction becomes more challenging with fog of war. The accuracy of all



(a) without Fog of War

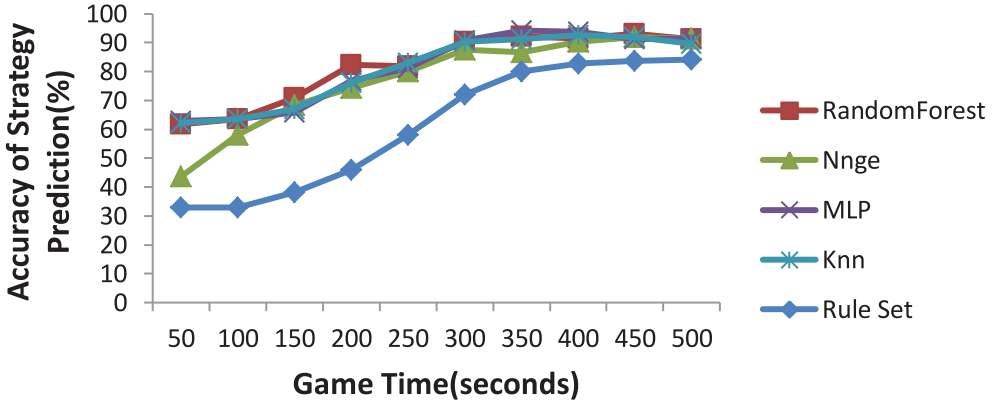


(b) with Fog of War

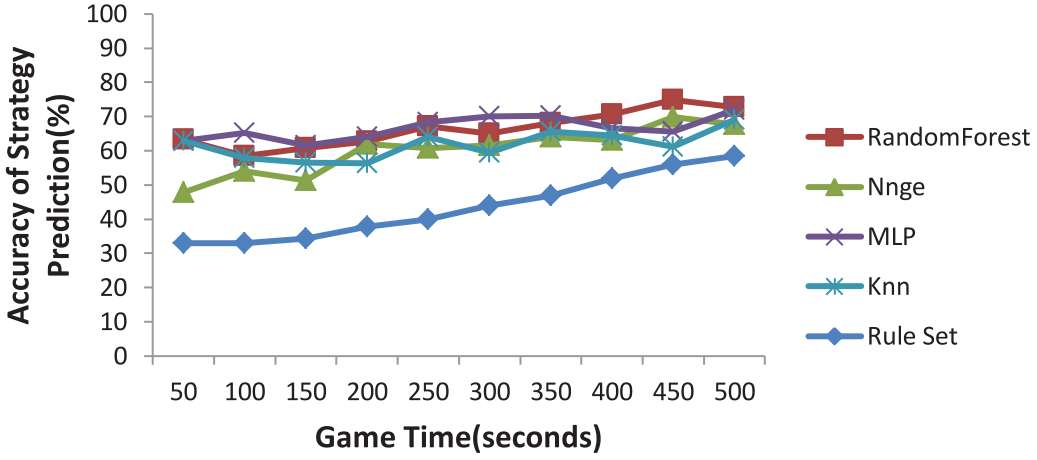
Fig. 7. Accuracy of strategy prediction for human vs. human games.

algorithms increases after approximately 150s. Interestingly, after 350s, the accuracy of Rule Set is just 42.0%, which is less than the ZeroR and considerably lower than without fog of war. This indicates that most players cannot see the build order of their opponent, even though the order has already been determined. From 350 to 450s, in most games, the build order is already determined, and the accuracy of the machine-learning algorithms is greater than the accuracy of the Rule Set.

4.2.2. Bots Vs. Bots. Figure 8 shows the accuracy of strategy prediction in bots vs. bots games without fog of war. Compared with human vs. human games, all machine-learning algorithms (except for NNGE) had approximately 63% accuracy at 50s. This is because the ZeroR is 63% here. As time progressed, the accuracy steadily increased,



(a) without fog of war

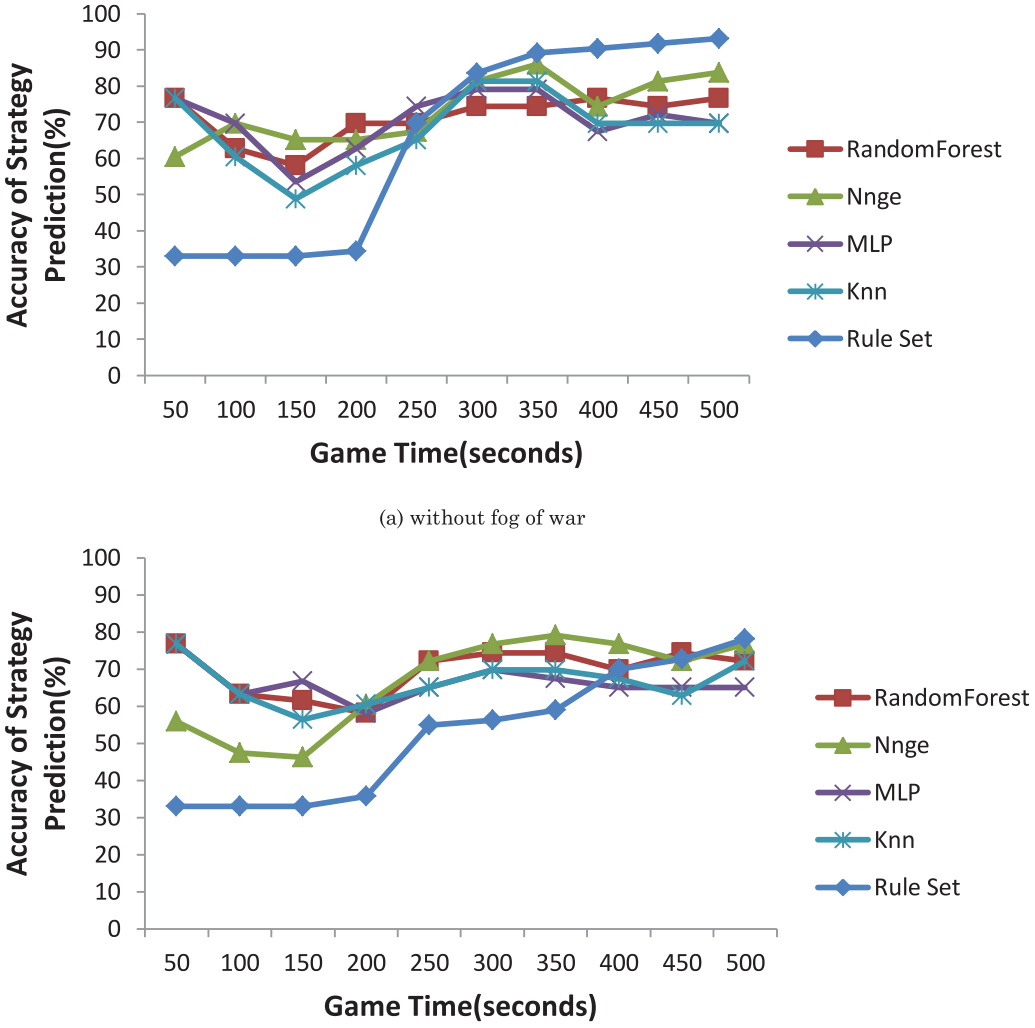


(b) with fog of war

Fig. 8. Accuracy of strategy prediction in bots vs. bots games.

reaching a slightly higher accuracy than that in human vs. human games. This is because most of the bots uses build orders that are not informed by the scouts. A second reason will be explained in detail in the Xelnaga vs. human section. With fog of war, accuracy increases more slowly, and the accuracy of the Rule Set in particular is very low. This is because most bots do not scout the opponent's base after the first scouting. However, the trends here are similar to those of the human vs. human games when fog of war is included.

4.2.3. Xelnaga vs. Human. Figure 9 shows the prediction accuracy of the human player's strategy in Xelnaga vs. human games without fog of war. The accuracy of the machine-learning algorithms is lower than the ZeroR or Rule Set. The accuracy of machine-learning algorithms decreases until 150s, and the accuracy of the Rule Set abruptly increases to 70% at 250s. This can be explained by considering that Xelnaga always chooses the build order Dark Templar, and, furthermore, Xelnaga builds a building named Citadel of Adun at an average of 200s after choosing Dark Templar. If the human players build Citadel of Adun before employing Dark Templar, then they



(a) without fog of war

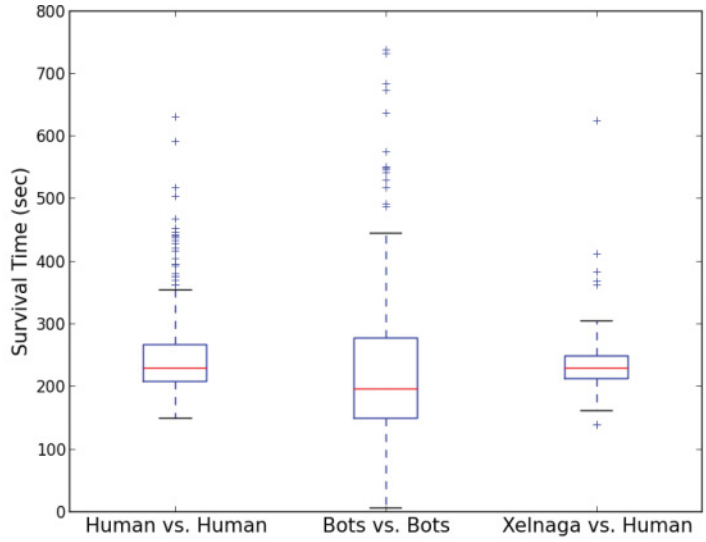
(b) with fog of war

Fig. 9. The accuracy of strategy prediction for human players in Xelnaga vs. human (the goal of learning is to predict human's strategy).

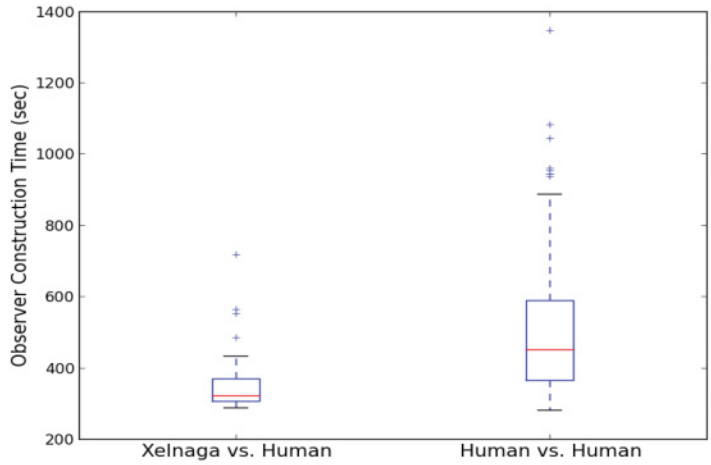
will hide this building. However, Xelnaga does not possess the skills necessary to do this, so most human players see Xelnaga's build order (Dark Templar) and change their build order to Observer. The results with fog of war are very similar to that without fog of war. In summary, is impossible to predict the strategy in case of Xelnaga vs. human because the human player can respond so well to scouting of Xelnaga's build order.

4.3. Summary of Results

Because of the fog-of-war, the most important element in predicting a strategy is the scout. In StarCraft, initial scouting is typically conducted before making air-borne scout units. Once scout units can be air borne, the survival time increases, and the unit



(a) A comparison of survival time of the first scouting units (in case of Xelnaga vs. Human, the scouting units belong to Xelnaga).



(b) Observer construction time for human players in Xelnaga vs. human and human vs. human games.

Fig. 10. Comparison of the statistics from different replay types.

can collect more information, so the accuracy of prediction increases. Figure 10 shows the survival time of the first scouting unit for the three types of games. For bots vs. bots games, the spread of survival time is very broad because some bots do not have an algorithm enabling them to kill scout units. Because of this, the average survival time is higher in bots vs. bots games. For the Xelnaga vs. human games, the graph shows a tight spread of survival times of Xenaga's scouting units because they always use the same algorithm without adaptation.

Figure 10 shows the time at which Observer construction is initiated by human players in Xelnaga vs. human and human vs. human games. This time is much shorter in the Xelnaga vs. human games because Xelnaga's Dark Templar construction time is

Table III. The accuracy (%) of Strategy Prediction Using the Machine-Learning Algorithm Random Forest and the Difference between the Rule Set and ZeroR

	"Fog of war"	Random Forest		Random Forest – Rule Set		Random Forest – ZeroR	
		250s	500s	250s	500s	250s	500s
Human vs. human	No	68.1	90.9	17.9	0.7	19.9	42.7
	Yes	56.4	67.4	18.4	−6.1	8.2	19.2
Bots vs. bots	No	82.0	91.2	24.0	7.0	18.7	27.9
	Yes	67.1	72.8	27.1	14.4	3.8	9.5
Xelnaga vs. human	No	69.8	76.7	−0.1	−16.5	−7.0	0.0
	Yes	72.1	72.1	17.1	−6.0	−4.6	−4.6

very fast. The human players are able to react to scouting data that reveal the build order of Xelnaga (Dark Templar) and respond accordingly by choosing Observer.

Table III shows the accuracy of strategy prediction at 250s and at 500s using the machine-learning algorithm Random Forest. Random Forest was the most effective in predicting the opponent's strategy.

5. CONCLUSIONS AND FUTURE WORKS

The prediction of an opponent's strategy in StarCraft is a challenging problem for AI. Because of the fog of war employed in the game, players have incomplete information about their opponents. We limited our study to Protoss vs. Protoss games and collected replay data from archives. The data were analyzed both with and without the fog-of-war effect, and human vs. human, bot vs. bot and Xelnaga vs. human replays were used for machine learning.

In the human vs. human and bot vs. bot games, machine learning was useful in predicting the opponent's strategy. The accuracy of the machine-learning algorithms was higher than 70% after 250s and was higher than the Rule Set and ZeroR without fog of war. When fog of war was included, the accuracy decreased, and the Rule Set performed considerably worse than the machine-learning algorithms. The difference in accuracy between the Rule Set and machine-learning algorithms was larger without fog of war. Specifically, the Rule Set was poor by the time the build order was established because of the incomplete information. In the Xelnaga vs. human replay data, Xelnaga always picked the same build order, which was Dark Templar. Because Xelnaga was not able to respond to the human player's scouting, the human players changed their build accordingly when they detected the build order of Xelnaga. In these data sets, strategy prediction failed both with and without fog of war.

Our research was limited to games involving only the Protoss race, and the three types of build order. This represents an area for expansion of our work into the different races and additional build orders. Additionally, the prediction accuracy was reduced when fog of war was included; improving the accuracy with fog of war is necessary. Furthermore, prediction is limited not only by the incomplete information available but also by a human player's prediction and reaction abilities. That is, human players were able to predict the opponent's strategy and change their build order accordingly during the Xelnaga vs. human games, and further research into how the human players were able to do this is warranted.

It is quite challenging to transfer the knowledge from academic research into the AI bots in competitions. Although we focus on the strategic prediction in this article, we also need to develop algorithms to change the build order adaptively based on the detected strategy.

REFERENCES

D. W. Aha, D. Kibler, and M. K. Albert. 1991. Instance-based learning algorithms. *Mach. Learn.* 6, 1, 37–66.

L. Breiman. 2001. Random forests. *Mach. Learn.* 45, 5–32.

M. Brent. 1995. *Instance-Based Learning: Nearest Neighbour with Generalisation*. Master's Thesis, University of Waikato, Hamilton, New Zealand.

M. Buro and D. Churchill. 2012. Real-time strategy game competitions. *AI Mag*. 33, 3, 106–108.

D. Churchill and M. Buro. 2012. Incorporating search algorithms into RTS Game agents. *AIIDE Workshop on Artificial Intelligence in Adversarial Real-Time Games*, 2–7.

IEEE CIG StarCraft AI Comeptition. 2011. Homepage. Retrieved from http://ls11-www.cs.uni-dortmund.de/rts-competition/starcraft-cig2011/.

IEEE CIG StarCraft AI Comeptition. 2012. Homepage. Retrieved from http://ls11-www.cs.uni-dortmund.de/rts-competition/starcraft-cig2012/.

E. Dereszynski, J. Hostetler, A. Fern, T. Dietterich, T. T. Hoang, and M. Udarbe. 2011. Learning probabilistic behavior models in real-time strategy games. In *Proceedings of the 7th Artificial Intelligence and Interactive Digital Entertainment Conference.* 20–25.

K. J. Kim and S. B. Cho. 2011. Server-side Early Detection Method for Detecting Abnormal Players of StarCraft. In *Proceedings of the 3rd International Conference on Internet*. 489–494.

F. Kabanza, P. Bellefeuille, F. Bisson, A. Benaskeur, and H. Irandoust. 2010. Opponent behavior recognition for real-time strategy games. In *Proceedings of the AAAI Workshop on Plan, Activity, and Intent Recognition*, 29–36.

K. J. Kim and S. B. Cho. 2012. The 2011 IEEE conference on computational intelligence and games report. *IEEE Comput. Intell. Mag.* 7, 1, 15–18.

J. Kim, K. H. Yoon, T. Yoon, and J. H. Lee. 2010. Cooperative learning by replay files in real-time strategy game. *Lecture Notes in Computer Science*, Vol. 6240. 47–51.

J. L. Hsieh and C. T. Sun. 2008. Building a player strategy model by analyzing replays of real-time strategy games. In *Proceedings of the IEEE International Joint Conference on Neural Networks.* 3106–3111.

Lord Martin's Replay Browser. 2012.

H. S. Park, H. C. Cho, K. Y. Lee, and K. J. Kim. 2012. Prediction of early stage opponent strategy for StarCraft AI using scouting and machine learning. In *Proceedings of the Workshop at SIGGRAPH ASIA (Computer Gaming Track).* 7–12.

AIIDE StarCraft AI Competition. 2010. Homepage. Retrieved from http://eis.ucsc.edu/StarCraftAICompetition/.

G. Synnaeve and P. Besiere. 2011a. A Bayesian model for RTS unit control applied to StarCraft. In *Proceedings of the IEEE Conference on Computational Intelligence and Games.* 190–196.

G. Synnaeve and P. Besiere. 2011b. A Bayesian model for opening prediction in RTS games with application to StarCraft. In *Proceedings of the IEEE Conference on Computational Intelligence and Games.* 281–288.

B. Weber and M. Mateas. 2009. A data mining approach to strategy prediction. In *Proceedings of the IEEE Symposium on Computational Intelligence in Games*. 140–147.

B. G. Weber and S. Ontanon. 2010. Using automated replay annotation for case-based planning in games. In *Proceedings of the International Conference on Case-Based Reasoning Workshop on CBR for Computer Games.* 15–24.

B. G. Weber, M. Mateas, and A. Jhala. 2011. Building human-level ai for real-time strategy games. In *Proceedings of the AAAI Fall Symposium on Advances in Cognitive Systems.* 329–336.

I. H. Witten, E. Frank, and M. A. Hall. 2011. *Data Mining: Practical Machine Learning Tools and Techniques* (3rd ed.). Morgan Kaufmann.

Received March 2013; revised May 2013; accepted July 2013