Proceedings of the Federated Conference on Computer Science and Information Systems pp. 391–399
DOI: 10.15439/2020F178
ISSN 2300-5963 ACSIS, Vol. 21

StarCraft agent strategic training on a large human versus human game replay dataset

Štefan Krištofík, Peter Malík
Institute of Informatics, Slovak Academy of Sciences
Dúbravská cesta 9, 845 07 Bratislava, Slovakia
Email: stefan.kristofik@savba.sk

Matúš Kasáš, Štefan Neupauer
Faculty of Informatics and Information Technologies
Slovak University of Technology
Ilkovičova 2, 842 16 Bratislava, Slovakia

***Abstract*—Real-time strategy games are currently very popular as a testbed for AI research and education. StarCraft: Brood War (SC:BW) is one of such games. Recently, a new large, unlabeled human versus human SC:BW game replay dataset called STARDATA was published. This paper aims to prove that the player strategy diversity requirement of the dataset is met, i.e., that the diversity of player strategies in STARDATA replays is of sufficient quality. To this end, we built a competitive SC:BW agent from scratch and trained its strategic decision making process on STARDATA. The results show that in the current state of the competitive environment the agent is capable of keeping a stable rating and a decent win rate over a longer period of time. It also performs better than our other, simple rule-based agent. Therefore, we conclude that the strategy diversity requirement of STARDATA is met.**

I. Introduction

REAL-time strategy (RTS) games are considered a very hard challenge for AI today. In an RTS game, players gather resources which they then use to build production facilities and military units with the goal to attack the opponent and destroy all of their structures. Compared to turn-based board games such as Chess or Go, the main challenges in RTS are partial observability of the game state and a huge complexity. RTS AI agents have to tackle the problems of decision making under uncertainty, because the opponent's intents are not always visible and known. A player can only see parts of a map near his armies and buildings, the rest is obscured by the fog-of-war. Players have to actively scout for the opponent's activity or utilize the domain knowledge to make some kind of partially informed decisions. The state space and the number of possible actions at each decision cycle for a player is very large. This makes it impossible to directly apply the techniques used in board games to RTS [3].

Since the 2003 call for research in the field of RTS game AI [1] there have been notable advancements in this area. Multiple approaches were explored and tried in the context of RTS AI. Efficient solutions to the problems posed by the RTS game environment can help not only in the rapidly growing video gaming industry by providing players better, more challenging and more rewarding experience, but also in other AI disciplines and domains of our lives. Weather forecasts, road traffic and self-driving cars, finance, personal assistants, or robotics are some examples of such complex dynamic environments where fast real-time decisions with incomplete information are required from agents [3], [5]. This area has attracted not only individual enthusiasts or researchers, but also teams from some of the big commercial companies like Facebook, Microsoft and Google [4].

This work is supported by Slovak national project VEGA 2/0155/19.

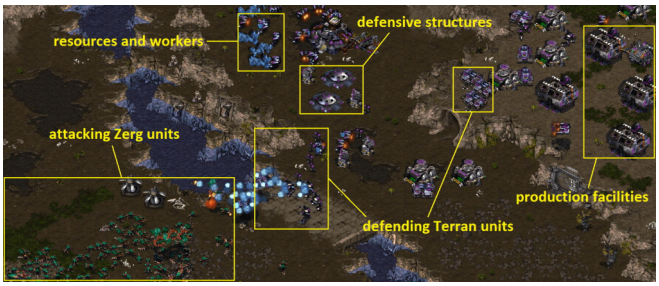

Fig. 1. StarCraft: Brood War

TABLE I
StarCraft: Brood War races

Race	Characteristics	Advantages and Disadvantages
Zerg	cheap units fast production fast expansion	strong early game hard army micromanagement slow regeneration
Terran	can repair slow expansion can build anywhere	strong defense hard special ability micromanag. needs most space for buildings
Protoss	expensive units slow production cannot repair	strong units with shields strong unit abilities buildings can be disabled

Currently, the most popular RTS game in the context of AI research is StarCraft: Brood War (SC:BW). Fig. 1 shows a screenshot of the game. It is universally praised for a very good balance of all three playable races: Terran, Zerg and Protoss. See Table I for a brief comparison of races. Although the game is now fairly old, released in 1998, both the competitive and research scenes are still very active. Development of SC:BW agents is well supported by the API called BWAPI [1] first introduced in 2009 as well as other useful tools.

It should be noted that the sequel to SC:BW, StarCraft II, has also been gaining popularity in the AI research community

[1]https://github.com/bwapi/bwapi

since the 2017 release of the game's API for AI research [2]. Although some very interesting results were achieved, e.g., by the Google Deep Mind team [5], the AI is not yet able to compete at a champion level [3].

Similarly to other AI disciplines such as image classification [9] or object detection [4], a number of challenges and tournaments are organized each year to compare the results of SC:BW agents, e.g., "Student StarCraft AI Tournament and Ladder (SSCAIT) [5], "BASIL Ladder" [6], "AIIDE StarCraft AI Competition" [7], "IEEE CoG StarCraft AI Competition" [8]. SC:BW AI agents are currently not yet able to reliably defeat expert level human players and even sometimes struggle against lower tier players as shown in a recent competition [4]. After beating humans in board games such as Chess and Go, overcoming human expert players in a genre of RTS games can be seen as the next goal for SC:BW AI agent research.

Throughout the game's long lifespan, a vast amount of SC:BW game replay data was accumulated and is available for players or AI agents to learn and improve. However, these data are scattered between many sources with various levels of quality. For a dataset to be a viable base for learning models for AI agents, it should be standardized so that it contains only valid replay files from the same game version, ideally containing games of a competitive nature between highly skilled players. Also it should contain replays of all 9 possible race match-ups, wide variety of competitive terrain maps, strategies and game lengths. An efficient usage of the replay data knowledge for training AI agents is still an open problem. Recently, a new large SC:BW replay dataset called STARDATA containing 65646 unlabeled human versus human replays was published by the Facebook research team [2]. The authors kept in mind many of the above mentioned requirements so this dataset is the first to be of sufficient size and quality for the task of AI agent training. It is the largest dataset of its kind to date. Prior work on compiling similar datasets resulted in much smaller sets of up to 7649 replays [2], [6], [7], [8]. Our goal in this work is to validate the strategy diversity requirement of STARDATA, which was not done yet. Also, to our knowledge, there were no attempts to use this dataset directly for learning.

The main contributions of this work are as follows. We validate the strategy diversity requirement of the STARDATA dataset containing SC:BW game replays. To this end, we build a competitive SC:BW Terran agent from scratch and train its strategic decision making process using solely the data extracted from the dataset. We test the agent's prowess in a continuous competitive environment (SSCAIT tournament) against other agents. The agent's win rate is evaluated. Also the ability to choose and execute a good counter-strategy to beat the opponent is inspected. Results show that the agent was able to maintain a stable competitive rating over a period of 4 months. This proves that an agent trained using strategy variety offered by STARDATA can be a strong contestant in the current state of the competition which in turn proves the strategy diversity requirement of the dataset is met. The results also show the Terran agent is performing better than our other, simple rule-based Zerg agent built from scratch trained using knowledge gained from a small machine versus machine replay dataset.

[2]https://news.blizzard.com/en-us/starcraft2/20944009/the-starcraft-ii-api-has-arrived

[3]https://www.nature.com/articles/d41586-019-03298-6

[4]https://www.kaggle.com/c/global-wheat-detection

[5]https://sscaitournament.com

[6]https://basil.bytekeeper.org

[7]http://www.cs.mun.ca/~dchurchill/starcraftaicomp/index.shtml

[8]https://cilab.gist.ac.kr/sc_competition

II. Related Work

Numerous attempts at strategy extraction or prediction from SC:BW replay datasets were conducted. In [6], the authors extracted 6 strategies for each race from a dataset containing 5493 replays. However, these strategies are not specifically targeted for different match-ups and are considered for all match-ups. Also, the extracted strategies are only considering low level units and only short games of up to 10 minutes. For strategy prediction, several machine learning algorithms were compared with promising results. In [7], the authors expand upon [6] by adding 570 new replays to the dataset. They also try to account for the fog-of-war information, i.e., the information about the areas that are currently not visible to the players. According to the results, the prediction model is not good for early game, but has better results for middle game predictions. In [8], the authors present a slightly larger dataset containing 7649 replays. Although some interesting statistical information was extracted from the dataset, the focus is rather on tactical aspects of the game, such as individual battle outcome prediction or battle detection.

In terms of dataset volume, STARDATA [2] is the largest one yet available for the research community. According to the results, it meets many requirements for a good base for learning models, such as match-up, map or game length diversity. However, the strategy diversity requirement is not addressed in detail. An attempt is also made at strategy extraction, but only few Protoss strategies were considered. Authors list several tasks that can be tackled by using their dataset. This work focuses on one of the tasks: strategy classification.

III. STARDATA training

A typical competitive SC:BW agent has 3 main modules:

- Macromanagement: involves long-term goals such as strategy, build order, unit compositions (see III-C2 for details) or expansion.
- Micromanagement: involves short-term goals such as combat effectiveness, unit positioning, targeting and behavior; also building placement.
- Scouting: involves gathering information about the opponent. It is often very important for the macromanagement module's decisions.

The quality of the agent depends on its ability to scout and identify the opponent's strategy and then to choose and effectively execute the proper counter-strategy to defeat them.

TABLE II
SC:BW STRATEGIES AND MATCH-UPS

Race	Terran	Protoss	Zerg	**In this work**
Terran	T (TvT)	TP (TvP)	TZ (TvZ)	yes
Protoss	PT (PvT)	P (PvP)	PZ (PvZ)	no
Zerg	ZT (ZvT)	ZP (ZvP)	Z (ZvZ)	

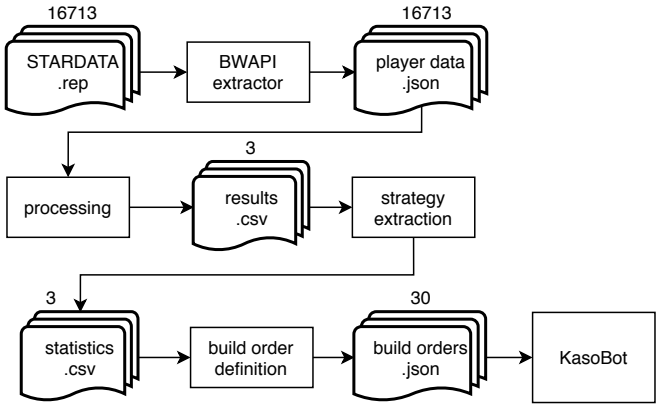

Fig. 2. Agent learning process

We built a competitive SC:BW Terran agent, nicknamed KasoBot, from scratch with the goal to prepare its strategic reasoning module before the game [9] by learning from unlabeled replay data from STARDATA [2]. We evaluate the agent from the strategic perspective, i.e., if it is able to select and execute the proper strategy in each game. We do not focus on the micromanagement tasks and only tackle this area partially (see III-F for details). Main goals of this work can be summarized as follows:

- Identify and categorize the most common player strategies from a subset of the STARDATA dataset.
- Label the strategies used by both players in this subset.
- Compare the strategies against each other in terms of win rate percentages.
- Build a competitive SC:BW agent able to execute the identified (extracted) strategies and, during a game, select the ones with a highest chance of winning based on the results of the comparison.
- Validate the strategy diversity requirement of STARDATA by evaluating the agent's results in a competitive environment.

Table II shows the strategy labels for each match-up which will be used in the remainder of this paper. In this work, we describe the extracted Terran strategies T, TP and TZ in TvT, TvP and TvZ match-ups, respectively. Our agent's learning process is shown in Fig. 2. The following sections will describe the individual steps.

TABLE III
DATASET

Replays	**Amount**
STARDATA Total	65646 [2]
Valid (BWAPI compatible)	65645
and competitive (2 players)	64550
and involving Terran	33741
and of length <15 minutes	**16713**
Versus Terran (TvT)	1468
Versus Protoss (TvP)	5014
Versus Zerg (TvZ)	10231

A. Dataset

We use STARDATA [2] for strategic learning. Several filters were applied before learning (Table III). Only replays that were executable in BWAPI, had exactly 2 active players, involved the Terran race on either side and had game length less than 15 minutes were considered. 16713 replays have passed these filters and were used for learning. The most numerous match-up was TvZ and the least numerous was the mirror match-up.

The length threshold was chosen because the diversity of late game strategies, i.e., strategies used in longer games, is worse than in early game. In other words, one can find less varied unit compositions used by players when inspecting later game stages than in early or middle stages. This is a consequence of the tech trees used in SC:BW. A player is required to first unlock the ability to build stronger, more expensive units, which takes some time. Therefore, these units typically appear in game only after a certain time has elapsed. We only want to learn early and middle game strategies. Late game strategies are easier to learn by manual inspection of some long games and analyzing the tech trees.

B. Raw data extraction

First, we extract player information from the original STARDATA replay files *(*.rep)*. For this purpose, we have implemented our own replay data extractor. This is a common practice in similar works [8], [2] due to the proprietary format of the original replay files that is hard to use directly. Each replay file is processed by running it in the game engine launched through the BWAPI interface together with our extractor. The following information is extracted from each replay file and stored in a separate *json* file:

- Basic information about the game: map name, length [10], player names and races, information needed to determine the winner (see III-C1 for details).
- For each building type used: name, ID [11], #built, #destroyed.
- For each building instance: timestamp [12] and current unit

[9] Other possible approaches are learning during the game or after the game [3]

[10] Measured in game frames. Competitive SC:BW games are played at 23.81 frames per second.

[11] BWAPI unique ID

[12] In frames

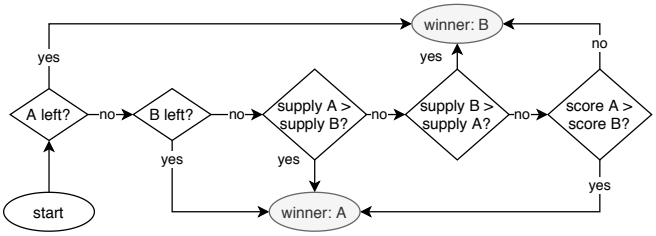

Fig. 3. Determining a winner of a SC:BW game. A, B - opposing players.

supply value [13] when it was built or destroyed.
- For each unit type used: name, ID, timestamp and current unit supply value when the first unit of that type was created, #built, #killed.
- For each finished tech or upgrade: name, ID, timestamp and current unit supply value when finished.

The above information is too generic and has to be processed further for strategy extraction.

C. Data processing

Next step is to process raw extracted player data into a more useful form as feature vectors. The results are stored in 3 *csv* files, one for each match-up that we are interested in: TvT, TvP and TvZ. The information about each TvP game is encoded on 1 row of the corresponding file. Same is true for TvZ games. However, the information about each TvT game is encoded on 2 rows (1 row from the perspective of each of the two players). After processing, the following information is available for each game:

- Basic information: similar to raw data (see III-B), but now also including the index of a winning player.
- For each building type used: timestamp and current supply value when the first instance of this type was constructed, maximum count of existing instances during the game.
- For each unit type used: #built.
- For each finished tech or upgrade: timestamp and current unit supply value when finished.
- For each of the selected building features from Table IV: building order (see III-C2 for details).
- For each of the selected unit features from Table IV: unit frequency (see III-C2 for details).

1) Determining a winner: To evaluate strategy win rates, it is necessary to determine the winning player in each game. This information is not explicitly available from the original replay files and must be gathered manually during processing. This process is illustrated in Fig. 3. If for any reason a player leaves the game, this action may be captured in the replay file as a player command and extracted. If such command is found, we always consider their opponent as a winner. Otherwise we consider the player with the largest unit supply value at the end of the game as a winner. This value is commonly used for this purpose [2]. If both players have the same unit supply values, we consider the player with the highest score as a winner. Score is an internal metric of SC:BW based on various player actions and achievements throughout the game.

2) Build orders and army compositions: SC:BW strategies are defined mainly by build orders and army compositions.

Build order refers to a specific sequence of building construction. As mentioned earlier, SC:BW uses tech trees. For example, to be able to create Marines, the player needs to construct Barracks, but Medics require both Barracks and Academy. Moreover, Academy can only be constructed if Barracks are already present. Therefore, the build order required to create Medics is: Barracks $\rightarrow$ Academy.

Army composition refers to a list of unit types which form the backbone of the player's army. In other words, army composition is defined by the most used unit types. Other, less frequently used unit types, should only be considered as support units for the main army composition. For example, one of the more popular Terran army compositions against Zerg opponents is Marines with Medics as the backbone with the support of few Siege Tanks and Science Vessels [14].

Our goal is to identify and categorize the most used strategies from STARDATA. Since our agent is Terran, we are interested in Terran strategies against all three races and Protoss and Zerg strategies against Terran. We have selected a set of the most important features for each race which will be used to distinguish between various strategies. The complete list is shown in Table IV. Other features not included in the table were deemed not as important for strategy characterization. We did not include mandatory buildings or units, i.e., used in every game because they are required to make any progress in the game whatsoever (e.g., Terran Barracks, Protoss Gateway, worker units). We also did not include some unpopular units which are built only very rarely (e.g., Protoss Scout, Zerg Devourer). We also did not include defensive structures, special non-combat units (e.g., Zerg Larva) and some late-game structures as well since we are interested in only early and middle game strategies.

To encode a build order into a feature vector, we assign each of the selected building features a number based on the order in which it was first constructed during a game. Buildings which were not built in a game are assigned $number\_of\_building\_features + 1$. In case of Terran, it is the value 9, since there are 8 building features. An example of a Terran build order is shown in Table V. The player has built Factory as first, followed by Machine Shop and has not built any Control Towers, Engineering Bays or Starports.

Based on the relative frequency of unit creation during a game, we assign each unit feature from Table IV a number representing how many units of this type were created relative to other types. We define the unit creation frequency as the number of units created per minute (1429 frames) starting since the frame when all the requirements for it were first satisfied during a game. Units created most frequently are

[13] Represents the current size of the player's army.

[14] E.g., https://www.youtube.com/watch?v=qyixL9J7-B8 at around 10 minute mark.

TABLE IV
SELECTED STRATEGY DEFINING FEATURES

Race	Buildings	Units
Terran	Academy Armory Command Center Control Tower Engineering Bay Factory Machine Shop Starport	Firebat Goliath Marine Medic Siege Tank Vulture Wraith
Protoss	Forge Nexus Robotics Facility Stargate Templar Archives	Archon Carrier Dark Templar Dragoon High Templar Zealot
Zerg	Hive Hydralisk Den Lair Spire	Hydralisk Lurker Mutalisk Scourge Ultralisk Zergling

TABLE V
EXAMPLE OF A TERRAN BUILD ORDER

Building	Build order	Building	Build order
Academy	5	Engineering Bay	9
Armory	4	Factory	1
Command Center	3	Machine Shop	2
Control Tower	9	Starport	9

assigned smaller numbers. Units which were not created in a game are assigned $number\_of\_unit\_features+1$. In case of Terran, it is the value 8, since there are 7 unit features. This way a feature vector of unit frequencies is formed. An example of a Terran unit frequency is shown in Table VI. The player has built many Siege Tanks and Vultures and has not built any Goliaths, Medics or Firebats.

D. Strategy extraction

We extract Terran strategies (from both player perspectives) from TvT, both Terran and Protoss strategies from TvP and both Terran and Zerg strategies from TvZ. Terran strategies against different races may be very different, so we treat each match-up separately.

We treat the STARDATA replays as unlabeled data because the particular strategies used by both opponents are not known.

TABLE VI
EXAMPLE OF A TERRAN UNIT FREQUENCY ANALYSIS

Unit	Frequency rating	Unit	Frequency rating
Marine	4	Wraith	3
Vulture	2	Medic	8
Goliath	8	Firebat	8
Siege Tank	1	-	-

To search and find regularities in unlabeled data, we have chosen a cluster analysis method. In particular, we used the K-Means clustering algorithm for strategy extraction. This algorithm requires to know the desired number of clusters beforehand. Each cluster represents one distinct strategy. After few experiments with the number of clusters ranging from 6 up to 20, it was selected to be 10 for each match-up. Using this number of clusters guaranteed their sufficient diversity as well as the sufficient number of replays in each cluster. The algorithm produces clusters of different sizes. This is beneficial because popularity of strategies can vary. Therefore, more popular strategies will be represented by larger clusters than the less popular ones.

This resulted in a successful extraction of a total of 30 different Terran strategies, 10 for TvT, 10 for TvP and 10 for TvZ. The information is stored in 3 *csv* files, one for each match-up. Moreover, 10 Protoss strategies for PvT and 10 Zerg strategies for ZvT were also extracted. However, as can be seen from Table II, this work does not provide details on them and only focuses on Terran strategies.

E. Results

The summary of extracted Terran strategies against all three races is shown in Fig. 4. Clusters representing strategies for each match-up produced by K-Means were initially labeled by the algorithm as 0-9. We assigned more descriptive labels to strategies to clearly indicate the relevant match-up (see also Table II), e.g., cluster 4 from TvZ was assigned label TZ4. The table shows average building orders and average unit frequencies with respect to corresponding clusters. This means that, for example, strategy T6 can be characterized by the building order starting very often with Factory, then usually following with Machine Shop or Starport, and very often ending with Armory. This strategy is also characterized by the unit composition containing Siege Tanks very often, Wraiths and Marines often, too, and other units only very rarely. Strategy descriptions use abbreviated unit names (e.g., G=Goliath). The term *expansion* means the construction of Command Center, effectively establishing another base to boost the economy. The term *mech* refers to mechanical units (Vulture, Goliath, Siege Tank and Wraith), *bio* refers to biological units (Marine, Medic and Firebat) and *combo* refers to a combination of these. From the results, the following important conclusions can be drawn:

- Two main Terran army composition types are prevalent: bio-based and mech-based. The combination of both is rarely used.
- Bio units are often used against Zerg, but rarely against Terran or Protoss.
- Mech units are often used against Terran and Protoss, but not often against Zerg.
- Combo unit composition is rarely used against Protoss and Zerg and almost never against Terran.

The extracted strategies seem to offer a good variety of build orders and unit compositions overall. In some cases, the differences between particular strategies are negligible in unit compositions, but significant in build order (e.g., compare

strategy	count	Academy	Armory	Command Center	Control Tower	Engineering Bay	Factory	Machine Shop	Starport	Marine (M)	Vulture (V)	Goliath (G)	Siege Tank (S)	Wraith (W)	Medic (E)	Firebat (F)	TvT strategy description
T0	111	7.58	2.92	6.41	8.53	7.54	1.05	4.53	7.74	3.06	2.26	1.89	7.05	7.81	8.00	7.95	mech GV, few SW, slow expansion
T1	423	7.16	8.05	4.67	5.66	7.88	1.10	3.32	2.76	3.49	2.31	7.98	2.47	2.52	7.97	7.97	mech VS with W
T2	814	5.73	3.90	2.83	7.95	7.04	1.12	2.60	6.89	3.91	2.48	1.99	1.72	8.00	7.99	7.96	mech SGV, fast expansion
T3	108	8.35	8.95	7.83	8.97	8.91	6.31	9.00	8.82	2.52	7.07	8.00	8.00	8.00	7.51	7.95	bio M, few VE, slow expansion
T4	373	5.65	4.36	2.64	7.45	7.17	1.19	2.80	5.43	4.80	2.50	2.83	1.97	3.23	8.00	7.98	mech SVG with W, fast expansion
T5	154	6.77	4.25	2.69	8.65	7.87	1.18	2.62	8.03	2.72	1.83	8.00	2.53	7.95	8.00	8.00	mech VS with M, fast expansion
T6	163	6.86	8.12	3.63	6.13	7.28	1.38	2.97	3.30	2.72	7.98	7.91	1.60	2.32	7.83	7.93	mech S with W, few bio
T7	234	6.30	4.06	3.15	7.67	6.87	1.27	2.43	6.47	3.16	8.00	1.63	1.50	7.29	8.00	8.00	mech SG, few W
T8	371	6.76	5.81	4.85	4.88	7.26	1.06	3.12	2.75	4.30	4.32	1.57	1.87	4.51	8.00	7.95	mech GS with VW
T9	185	8.28	8.96	4.37	8.90	7.69	1.10	2.43	8.51	2.51	2.03	8.00	3.20	8.00	8.00	8.00	mech VS with M
																	TvP strategy description
TP0	524	9.00	8.72	3.02	8.99	5.32	1.08	2.16	8.81	2.47	1.40	8.00	2.26	7.95	8.00	8.00	mech VS with M, expansion
TP1	1454	5.33	5.69	2.77	8.05	4.43	1.19	2.35	6.99	3.98	1.29	2.50	2.35	7.80	7.97	7.99	mech VSG, few W, bio
TP2	811	5.20	5.66	2.83	8.67	4.20	1.18	2.28	7.91	3.04	1.09	8.00	1.97	7.91	7.91	7.97	mech VS, few W, bio
TP3	199	8.19	8.70	2.17	8.90	5.47	1.44	2.78	8.75	1.91	8.00	8.00	1.66	7.90	8.00	8.00	combo SM, few W, fast expansion
TP4	252	8.65	8.95	9.00	9.00	9.00	1.00	2.54	8.81	1.99	2.42	8.00	3.44	7.98	8.00	7.98	combo MVS, few W, no expansion
TP5	163	7.77	9.00	7.39	9.00	8.52	7.23	9.00	9.00	1.65	8.00	8.00	8.00	8.00	7.54	7.79	bio M, few EF, slow expansion
TP6	625	7.30	8.16	5.02	4.87	5.97	1.03	2.21	3.39	3.21	1.25	6.97	2.23	6.54	7.96	7.97	mech VS with WG, few bio
TP7	152	2.88	8.66	3.75	8.53	5.03	2.12	3.76	8.04	1.14	6.94	7.91	3.11	7.84	3.35	6.89	bio ME with S, few FV
TP8	172	7.97	8.83	9.00	8.96	3.01	1.03	2.06	8.81	2.51	1.89	7.97	2.31	7.98	7.95	8.00	mech VS with M, no expansion, fast ebay
TP9	662	4.60	8.91	2.93	8.59	4.31	1.17	2.33	8.13	2.86	1.10	8.00	2.16	7.88	7.85	7.95	mech VS with M, expansion, academy
																	TvZ strategy description
TZ0	1126	6.50	3.73	2.98	8.48	5.25	1.38	3.14	7.77	3.30	3.05	1.75	3.63	7.93	7.48	7.92	mech G with VS, few bio
TZ1	2660	2.14	8.92	1.17	7.07	2.89	3.88	5.39	5.73	1.04	7.98	7.96	2.64	7.59	2.70	5.24	bio ME with SF, fast expansion
TZ2	1063	1.78	9.00	3.27	9.00	4.02	9.00	9.00	9.00	1.16	8.00	8.00	8.00	8.00	2.75	4.36	bio ME with F
TZ3	792	8.63	8.99	7.15	9.00	8.65	8.27	8.93	9.00	1.38	7.69	8.00	7.98	8.00	8.00	8.00	bio M, few V, slow expansion
TZ4	683	1.48	8.89	7.70	6.75	3.79	2.08	5.03	5.39	1.06	7.16	7.92	4.20	7.70	2.70	4.42	bio ME with SF, slow expansion
TZ5	423	5.44	8.73	7.15	5.19	6.42	1.08	7.52	2.20	1.63	3.59	7.92	7.98	2.76	4.94	7.52	bio M with WVE, slow expansion
TZ6	1134	2.44	8.60	1.16	7.00	2.99	3.70	5.49	5.66	1.16	2.71	7.68	4.18	7.59	3.32	5.23	bio M with VESF, fast expansion
TZ7	1339	2.20	8.87	1.25	7.66	2.85	3.85	8.00	6.37	1.03	7.91	7.98	8.00	7.55	2.21	5.15	bio ME with F, few W, fast expansion
TZ8	583	5.16	8.30	5.46	4.77	5.79	1.08	4.49	2.38	1.27	4.64	7.83	2.79	4.85	3.36	7.13	combo MS with EVW
TZ9	428	7.90	4.73	5.39	5.12	6.70	1.07	3.50	2.61	3.62	2.96	1.58	3.87	5.78	7.76	7.98	mech G with VSW, few bio
		building order (1=built first, 9=never built)								unit frequency (1=always, 8=never)							

Fig. 4. Extracted Terran strategies: versus Terran (top), versus Protoss (middle), versus Zerg (bottom) with average building orders (left), average unit frequencies (middle) and verbose descriptions (right); building order: 1=built as first, 9=never built; unit frequency: 1=always created, 8=never created

strategies TP0 and TP8). Also, the resulting sizes of clusters seem to be pretty varied, too, each containing a decent sample of at least 100 occurrences in match-ups. Strategy distributions in each match-up are shown in Fig. 5. For each match-up, there appears to exist one favorite strategy with a large margin before other strategies, e.g., TZ1 for TvZ.

Win rates of strategies from Fig. 4 are compared against each other in TvT in Fig. 6, against Protoss PvT strategies in Fig. 7 and against Zerg ZvT strategies in Fig. 8. The numbers of match-ups containing the exact pair of strategies is shown on the right sides of Figs. 6-8. For example, if in a TvP match-up the Protoss opponent is following strategy PT8, the best course of action for the Terran player is to choose strategy TP2, because it has the highest win rate against that enemy strategy (82.31 %). The number of match-ups involving these exact strategies was 147. The second best option would be to choose TP9 with a 77.62 % win rate. The diagonal of the TvT table is always 50 % because both players chose the same strategy and only one won. Note that some strategy match-ups never occurred (e.g., TZ3 versus ZT0). Win rates are not available in those cases. This learned data will be helpful for the agent strategic decision making during games. See III-F for details.

We analyze strategies further in Table VII. According to the results, in both TvT and TvP match-ups, the most played Terran strategy is not the best one (using weighted averages across all games where the strategy was involved). In the TvZ match-up, the most played Terran strategy is also the most successful one.

F. Agent function

All 30 extracted Terran strategies (see Fig. 4) are transformed into build orders and described in 30 *json* files, one for each strategy. The agent, KasoBot, is able to emulate all 30 strategies by reading the contents of these *json* files. At

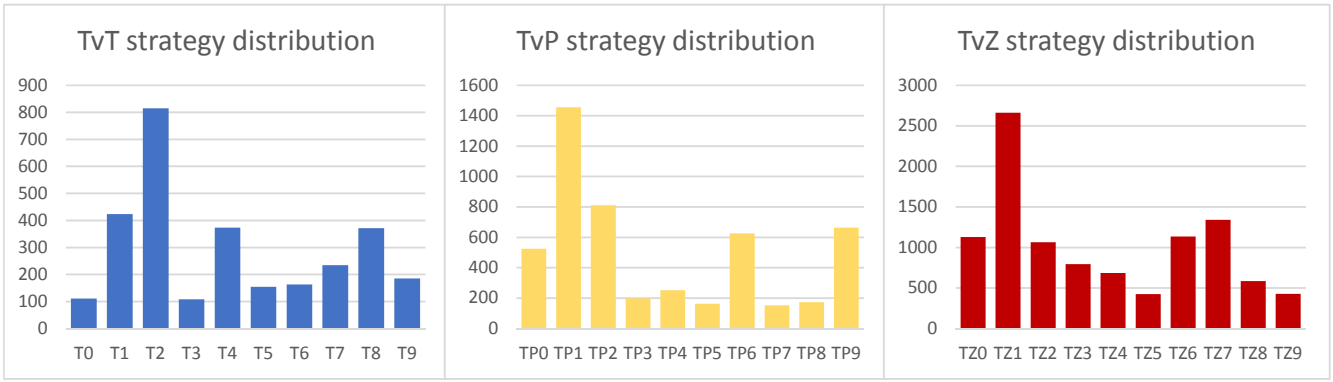

Fig. 5. Extracted strategy distributions: number of games the strategy has occurred in

Terran player B strategy

Terran player A strategy	T0	T1	T2	T3	T4	T5	T6	T7	T8	T9
T0	50.00	53.49	28.57	75.00	0.00	42.86	55.56	0.00	66.67	56.25
T1	46.51	50.00	42.41	100.00	46.34	75.00	68.42	43.24	50.00	80.00
T2	71.43	57.59	50.00	100.00	48.31	47.62	38.46	51.61	48.74	53.57
T3	25.00	0.00	0.00	50.00	N/A	100.00	50.00	N/A	0.00	0.00
T4	100.00	53.66	51.69	N/A	50.00	71.43	87.50	47.06	58.33	100.00
T5	57.14	25.00	52.38	0.00	28.57	50.00	36.36	75.00	N/A	61.54
T6	44.44	31.58	61.54	50.00	12.50	63.64	50.00	53.57	53.33	83.33
T7	100.00	56.76	48.39	N/A	52.94	25.00	46.43	50.00	57.50	77.78
T8	33.33	50.00	51.26	100.00	41.67	N/A	46.67	42.50	50.00	100.00
T9	43.75	20.00	46.43	100.00	0.00	38.46	16.67	22.22	0.00	50.00

Terran player B strategy

Terran player A strategy	T0	T1	T2	T3	T4	T5	T6	T7	T8	T9
T0	20	43	7	4	1	7	9	1	3	16
T1	43	38	158	5	41	20	19	37	42	20
T2	7	158	240	1	118	42	39	62	119	28
T3	4	5	1	88	0	1	2	0	1	6
T4	1	41	118	0	106	7	8	17	72	3
T5	7	20	42	1	7	36	11	4	0	26
T6	9	19	39	2	8	11	26	28	15	6
T7	1	37	62	0	17	4	28	36	40	9
T8	3	42	119	1	72	0	15	40	78	1
T9	16	20	28	6	3	26	6	9	1	70

Fig. 6. TvT: strategy win rates (left), strategy match-ups (right)

Protoss player strategy

Terran player strategy	PT0	PT1	PT2	PT3	PT4	PT5	PT6	PT7	PT8	PT9
TP0	30.77	59.34	30.36	50.00	29.73	14.29	33.33	0.00	47.20	19.05
TP1	63.38	61.54	56.84	60.00	53.72	80.00	53.57	N/A	70.77	50.35
TP2	56.72	58.82	46.25	87.50	67.57	55.56	37.63	N/A	82.31	41.57
TP3	33.33	11.43	0.00	7.14	11.11	23.08	0.00	100.00	19.44	0.00
TP4	0.00	43.94	25.00	34.33	25.00	57.14	N/A	52.38	33.82	0.00
TP5	N/A	29.63	N/A	23.53	33.33	40.00	50.00	34.33	20.00	N/A
TP6	54.55	47.83	58.00	60.71	60.00	50.00	47.83	0.00	67.48	38.18
TP7	72.73	50.00	66.67	27.27	61.11	50.00	32.43	0.00	64.52	50.00
TP8	33.33	52.94	33.33	54.17	46.15	55.56	50.00	100.00	54.00	0.00
TP9	60.19	80.00	46.67	78.57	58.78	54.29	25.53	N/A	77.62	36.73

Protoss player strategy

Terran player strategy	PT0	PT1	PT2	PT3	PT4	PT5	PT6	PT7	PT8	PT9
TP0	26	91	56	46	37	28	3	2	214	21
TP1	284	13	285	5	121	5	392	0	65	284
TP2	134	17	240	8	74	9	93	0	147	89
TP3	3	70	5	28	27	26	1	1	36	2
TP4	3	66	4	67	12	7	0	21	68	4
TP5	0	27	0	51	6	5	2	67	5	0
TP6	77	46	150	28	75	24	46	1	123	55
TP7	22	10	12	11	18	6	37	3	31	2
TP8	9	34	12	24	26	9	2	2	50	4
TP9	108	30	105	14	131	35	47	0	143	49

Fig. 7. TvP: strategy win rates (left), strategy match-ups (right)

Zerg player strategy

Terran player strategy	ZT0	ZT1	ZT2	ZT3	ZT4	ZT5	ZT6	ZT7	ZT8	ZT9
TZ0	72.22	67.50	50.00	65.43	48.14	50.00	31.71	37.04	51.04	65.00
TZ1	63.65	79.17	100.00	67.35	70.27	55.84	75.00	28.06	71.43	100.00
TZ2	14.29	20.00	74.07	25.00	37.74	7.81	29.72	0.00	63.04	64.14
TZ3	N/A	0.00	47.15	16.67	N/A	0.00	10.53	N/A	0.00	28.04
TZ4	47.45	61.11	82.35	50.00	46.88	34.02	46.03	13.33	82.76	79.31
TZ5	46.15	54.43	81.48	45.76	31.58	28.57	25.00	0.00	62.92	78.26
TZ6	60.81	44.44	100.00	60.00	50.00	65.00	40.00	21.71	58.33	50.00
TZ7	64.71	75.65	75.00	32.09	62.50	41.70	61.82	44.05	75.00	81.08
TZ8	59.76	56.25	100.00	62.35	55.36	50.63	50.00	25.93	57.97	60.00
TZ9	52.63	38.24	100.00	48.28	48.95	60.00	26.92	25.00	58.00	100.00

Zerg player strategy

Terran player strategy	ZT0	ZT1	ZT2	ZT3	ZT4	ZT5	ZT6	ZT7	ZT8	ZT9
TZ0	36	40	8	81	538	116	164	27	96	20
TZ1	1029	72	4	441	74	539	124	335	35	7
TZ2	7	10	81	92	53	64	471	2	46	237
TZ3	0	2	649	6	0	1	19	0	8	107
TZ4	137	36	17	150	32	97	126	30	29	29
TZ5	39	79	27	59	38	21	24	1	89	46
TZ6	518	9	1	105	12	100	25	350	12	2
TZ7	136	115	4	134	136	259	406	84	28	37
TZ8	164	16	1	162	56	79	4	27	69	5
TZ9	19	68	1	29	190	35	26	8	50	2

Fig. 8. TvZ: strategy win rates (left), strategy match-ups (right)

TABLE VII
STRATEGY ANALYSIS

Match-up	Most used	Average win rate	Highest avg. win rate
TvT	T2	50.86 %	T4 (54.16 %)
TvP	TP1	56.46 %	TP9 (58.31 %)
TvZ	TZ1	59.59 %	TZ1

the start of a game, it randomly selects one of the 3 best performing strategies to follow against the opponent, using data from Figs 6-8. Opponent's race is known prior to the game. It proceeds to create buildings according to the extracted average build orders with the goal to reach the corresponding target unit composition.

The agent sets its highest priority to complete the required buildings from the selected build order. Unit production has lower priority and is only launched when the required buildings are ready. Once the build order is completed, the agent starts to produce units according to the selected unit composition. If existing production facilities are busy and there are spare resources available, the agent adds more production facilities to speed up the unit production. During a game, it periodically checks the availability of all buildings from the build order and tries to reconstruct them if any were destroyed.

Once enough military units are produced, the agent starts to form individual unit squads with different tasks, including scouting (small squads), defending important positions (mainly expansions), or attacking multiple revealed opponent positions (mainly structures) simultaneously. If the opponent army is confronted, the units follow simple combat behavior (attack closest enemy), with some exceptions (e.g., Vulture, which employs hit-and-run tactics).

During a game, the agent is also able to switch between similar strategies and adjust its army composition slightly. For example, if using strategy TP2 and the enemy starts to create air units, the agent will add Goliaths (strong anti-air units) to its composition, effectively switching to a strategy very similar to TP1.

If a game progresses to the late stage (15 minutes), the agent lowers the priority for the current strategy and sets the highest priority to a special late game strategy, which was manually constructed specifically for late game scenarios. The agent will modify its army composition by constructing late game buildings and units not included in extracted strategies, e.g., Battlecruisers (overall strong air unit) or Valkyries (strong air-to-air support unit). It will also focus more on the weapon and armor upgrades, making units more powerful and tough. This late game strategy is a result of manual inspection of several longer games, where the Terran players' strategies seemed to converge towards one particular late game strong army composition.

Although the extracted strategies and unit compositions serve as a source for strategic decisions, the game of SC:BW encompasses many other tasks that are required to beat the opponent. These tasks include: producing enough worker units to keep the economy afloat, supporting unit production; producing defensive structures; choosing important areas of the map to scout, defend and attack; producing enough maximum unit supply increasing buildings to keep the army at the maximum possible size and strength; creating special defensive building formations to prevent the opponent access to certain areas. All these tasks are performed by the agent simultaneously with the strategy component described above.

TABLE VIII
STRATEGIES IN A RULE-BASED AGENT

Strategy	Brief description
LateGame	priority: Mutalisks, secondary: Hydralisks, Zerglings standard strategy used when it is late game
Mutalisk	priority: Mutalisks, secondary: Zerglings switch to LateGame if failed
Hydralisk	priority: Hydralisks, secondary: Zerglings switch to LateGame if failed
ZerglingRush	priority: Zerglings switch to Hydralisk or Mutalisk if failed

After the manual inspection of multiple games, we conclude that the agent is able to select a good starting strategy and switch between similar strategies, as mentioned above. However, it is lacking in combat scenarios against superior opponents, e.g., unit behavior in combat is very simple compared to other advanced agents. As mentioned earlier, we did not focus on the micromanagement tasks as much. Moreover, the agent is not very good at scouting in early game and instead relies heavily on the extracted statistics. These aspects could be improved as future work.

IV. RULE-BASED TRAINING

We manually analyzed 112 machine versus machine games between the top 16 contestants of the final tournament in the SSCAIT 2018/19 edition. Based on the results, we prepared the strategic reasoning module for the Zerg agent, nicknamed NuiBot and built from scratch, by defining a number of simple rules and defining a number of own and opponent strategies.

The agent has a total of 12 different strategies and is able to switch between them during a game if needed. It also has a total of 13 different opponent models and is able to assign a different model to an opponent during a game if updated information is available. The agent tries to actively scout the opponent during early and mid game and update the opponent model as frequently as possible. Some of the agent's strategies are listed in Table VIII and some of the opponent models are listed in Table IX.

Apart from the strategic decisions controlled by the defined rules, NuiBot also performs a number of additional tasks, similar to KasoBot. These tasks involve scouting using workers in early game, simple army movements, attacking enemy positions and so on. However, it can not create unit squads and attacks with large groups. Unit behavior in combat is also very simple, similar to KasoBot.

Additionally, the agent continually accumulates the information about each individual opponent it has met previously in a game. In particular, it stores statistical data about models

TABLE X
AGENT RESULTS IN SSCAIT TOURNAMENT AS OF JULY, 1ST 2020; ACCUMULATED OVER 4 MONTHS

Agent	Description	ELO rating	SSCAIT rank	Overall win rate	Last 50 games
KasoBot (Terran)	Trained on STARDATA	2004	D	27 %	48 %
NuiBot (Zerg)	rule-based	1915	E	29 %	32 %

TABLE IX
OPPONENT MODELS IN A RULE-BASED AGENT

Opponent model	Brief description
fastExpand	opponent has expansion early
CannonRush	opponent is Protoss has Forge, but not Gateway early
ZerglingRush	opponent is Zerg has Spawning Pool, but only 1 Hatchery early
Dark Templar	opponent is Protoss has Citadel of Adun early
hardDefense	opponent has high number of defensive structures
massFlights	opponent has more than 3 air units
Normal	default model used until switched to another model

assigned to opponents during games. It updates these statistics with each played game. This helps to set a correct strategy next time when it faces the same opponent, i.e., it makes assigning a model to an opponent easier during subsequent games.

V. EXPERIMENTS

Both agents, KasoBot and NuiBot, were placed into SSCAIT Ladder at the start of March 2020. In this competition format, the opponents are picked randomly from a roster of all other active contestants. The authors can upload new versions of agents anytime. Over the course of 4 months, the results were accumulated and are summarized in Table X and are valid as of July, 1st 2020 (https://sscaitournament.com/index.php?action=scores).

KasoBot was able to maintain a relatively stable average ELO rating of 2004 (current contestant ratings range from 1224 up to 2917) with a decent 27 % overall win rate and a good 48 % current win rate (from last 50 games) with a slightly below average 'D' SSCAIT competitive rank (current contestant rankings range from 'B' to 'E'). ELO rating is a common metric used in many games, e.g., chess. SSCAIT rank is an internal SSCAIT system used to qualitatively compare agents. In comparison, NuiBot achieved a lower ELO rating of 1915, 29 % overall win rate and 32 % current win rate. The higher overall win rate of the rule-based agent is caused by the known phenomenon where rule-based agents tend to have good win rates when freshly entering a competition, but are slowly surpassed over time by more advanced agents, as recently documented for example in the AIIDE 2017 competition [4]. The overall win rate of KasoBot is expected to raise above NuiBot's if its current win rate stays at the present level. We conclude that the STARDATA trained agent surpassed the simple rule-based agent in all important statistics of the competition.

VI. CONCLUSION

We evaluate the strategy diversity requirement of the recently published large human versus human SC:BW replay dataset called STARDATA. We show that a competitive SC:BW agent built from scratch, with its strategic decision making module trained solely on the unlabeled replay data from STARDATA, can be a strong contestant among other agents in a competitive environment. The dataset offers a good variety of player strategies and the agent was able to learn broad amount of domain knowledge from the replays alone. Therefore, we conclude that the diversity requirement of STARDATA is met. It is encouraged to use this dataset for further work in the areas and tasks outlined by the authors.

REFERENCES

[1] M. Buro, "Real-time strategy games: A new ai research challenge," *International Joint Conferences on Artificial Intelligence, IJCAI 2003,* pp. 1534-1535.

[2] Z. Lin, J. Gehring, V. Khalidov and G. Synnaeve, "STARDATA: A StarCraft AI Research Dataset," *13th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2017,* pp. 50–56, arXiv:1708.02139.

[3] S. Ontañon, G. Synnaeve, A. Uriarte, F. Richoux, D. Churchill and M. Preuss, "A Survey of Real-Time Strategy Game AI Research and Competition in StarCraft," *IEEE Transactions on Computational Intelligence and AI in games, IEEE Computational Intelligence Society, 2013,* 5(4), pp. 1–19, doi: 10.1109/TCIAIG.2013.2286295.

[4] Mi. Čertický, D. Churchill, K.-J. Kim, Ma. Čertický and R. Kelly, "StarCraft AI Competitions, Bots and Tournament Manager Software," *IEEE Transaction on Games, 2018,* 11(3), pp. 227–237, doi: 10.1109/TG.2018.2883499.

[5] O. Vinyals, I. Babuschkin et al., "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature, 2019,* 575, pp. 350–354, doi: 10.1038/s41586-019-1724-z.

[6] B. G. Weber and M. Mateas, "A data mining approach to strategy prediction," *IEEE Symposium on Computational Intelligence and Games, 2009,* pp. 140-147, doi: 10.1109/CIG.2009.5286483.

[7] H. C. Cho, K. J. Kim and S. B. Cho, "Replay-based strategy prediction and build order adaptation for StarCraft AI bots," *IEEE Conference on Computational Intelligence in Games (CIG), 2013,* pp. 1-7, doi: 10.1109/CIG.2013.6633666.

[8] G. Synnaeve and P. Bessière, "A Dataset for StarCraft AI & an Example of Armies Clustering," *Artificial Intelligence in Adversarial Real-Time Games, 2012,* arXiv:1211.4552.

[9] W. Gong, X. Zhang, B. Deng and X. Xu, "Palmprint Recognition Based on Convolutional Neural Network-Alexnet," *Federated Conference on Computer Science and Information Systems, FedCSIS 2019,* 18, ACSIS, pp. 313–316, doi: 10.15439/2019F248.