Web-Based Interface for Data Labeling in StarCraft

In-Chang Baek
Dept. of Computer Science and Engineering
Sejong University
Seoul, South Korea
bic4907@gmail.com

Kyung-Joong Kim*
Dept. of Computer Science and Engineering
Sejong University
Seoul, South Korea
kimkj@sejong.ac.kr

***Abstract*—Recently, StarCraft AI has been very actively researched, largely via analysis of human replay data. However, such data are difficult to evaluate visually because they represent information from a limited environment, that of the game client. To solve this problem, we created an environment in which game screens are displayed on the web, allowing game progression to be evaluated at a glance. This allows the performance of more diverse and efficient experiments than conventional human testing. We show that human players label macro decisions (e.g., main force operations) during supervised StarCraft learning using a web-based interface.**

***Index Terms*—human test, StarCraft AI, visualizer, replays**

I. Introduction

StarCraft, which was released in 1998 by Blizzard, is a game of three races. Many experts and optimized strategies are available. A large set of complex replays has attracted AI researchers. For example, the Facebook AI team released 300G of data extracted from about 65,000 games [1].

BWAPI can be used to hack StarCraft clients to obtain the information needed for a game (units, buildings, and statements) and to control certain units via code. BWAPI is also used for StarCraft AI development. StarCraft AI competitions are held every year.

Fig. 1. The screen of StarCraft

Compared to other video games released with the latest forms of learning reinforcement, StarCraft is of the Real Time Strategy (RTS) genre; various environments and conditions that greatly affect game flow are available. Especially, unit combinations and interactions determine the flow of the game and the difficulty of decision-making. In each situation, it is essential to read the flow to predict the future and make the optimal choice. The use of deep/reinforcement learning to make game decisions remains at an early stage of development [2]. In fact, even the existing bots choose StarCraft strategies using a finite state machine (FSM) [3]

Previous research has shown that macro decision-making requires human labeling (e.g., in what direction will the main force move?). The game can turn on the basis of that decision. Also, it is important to predict where the main enemy force is and what actions it will take. The AI model searches every frame for the main force and makes the best possible decision. The real question is as follows: 'Where is the main force and where will it move to in five seconds?' Professional players chose the five-second window. As viewing long games remains difficult, we sought to solve the problem by watching only one scene.

We built a Web-based environment because of the limited functionality of the StarCraft client. As shown in Figure 1, the client views the game screen locally, rendering it difficult for another to evaluate the situation at a glance. Although a mini-map is available, this is not very helpful because the unit type is not shown. In the original StarCraft replay (provided by Blizzard), it is not possible to rewind the game or jump to specific scenes or times. Because of these inconveniences, we created our Web environment. Previously, StarCraft AI performance was evaluated using human players [4]. However, the work was performed offline and time and space constraints were in play. Here, we sought to deal with these issues.

II. A Web-Based StarCraft Data Labeling System

Environments such as described here typically provide four functions (TABLE I. A single scene is studied, but it is possible to view past scenes when evaluating changes in successive scenes. Also, the small units in the resized map are now easily identified by color-coding, reducing the time required for evaluation.

We used JSON software to move StarCraft data to the Web environment. StarCraft does not support conversion of replay files to JSON, but BWAPI allows data-writing in any desired format. The data required here are:

- Map, player, and unit information for each frame (position, type, health level, shield level [a Protoss race property], and fog of war in game)

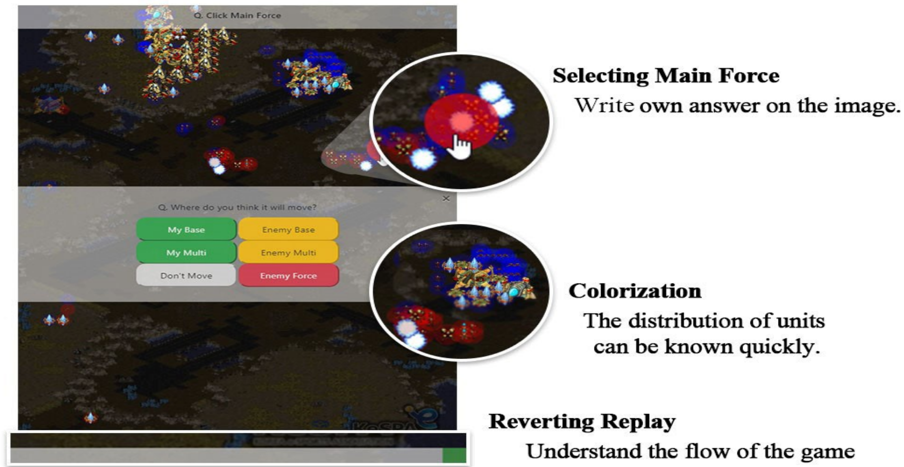

Fig. 2. Web Interface of Environment (https://cilab.sejong.ac.kr/sc/evalution)

This information defines only one scene and identifies only the main force, but it shows a second scene about 3 minutes into the future. The method is implemented in Python and extracts and compresses only the required data when the user solves the problem employing the JSON file. The data created are distributed via a Web socket. The server submits the problem to the user in real time after considering the submission situation. The interface creates an HTML5 canvas of the problem dataset. We constructed the game screen using available game data and implemented a function sending human answers to the server.

TABLE I. FUNCTIONS OF ENVIRONMENT

Function	***Description***
Rewind	Watch future scences like a video.
Colorization	Learn the distributions of units (allies, enemies).
Coordinate selection	Mark the position on the image.
Option selection	Additional questions can be posted.
Time measurement	Measuring the time it takes to solve a problem.

The system collects data from human players in the following format:

- Replay name, human tester ID, frame count (scene number), elapsed time, main force position, and decision.

The file name and frame count are used both to match the scene in question and for AI prediction. Performance is measured by the differences between the co-ordinates selected by the human and the AI, the main force-detection algorithm. If the positional difference is within a given error range, the behavioral performance can be scored.

III. CONCLUSION AND FUTURE WORKS

Unlike existing human tests [4], the human evaluator provides customized functions by moving from the limited environment of a game client to a Web-based environment. If the game client lacks the necessary interface for experimentation, or if it is felt that use of such an interface would waste time, our approach improves productivity (saving time and expanding knowledge). For example, not only is it possible to evaluate a given scene it is also possible to add an item and evaluate the previous scene, allowing interpretation of the experiment.

Our work was based on StarCraft I, but it is extensible to various interfaces and problem types as required. Our interfaces can be applied to study other RTS games where flow is important, including Warcraft and StarCraft II.

ACKNOWLEDGEMENTS

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (grant no. 2017R1A2B4002164).
*: corresponding author

REFERENCES

[1] Z. Lin, G. Jonas, K. Vasil, and G. Synnaeve, “STARDATA: A StarCraft AI Research Dataset,” AIIDE 2017.

[2] I. S. Oh and K. J. Kim, "Testing reliability of replay-based imitation for StarCraft," 2015 IEEE Conference on Computational Intelligence and Games (CIG), Tainan, 2015, pp. 536-537.

[3] B. G. Weber, P. Mawhorter, M. Mateas and A. Jhala, "Reactive planning idioms for multi-scale game AI," Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games, Dublin, 2010, pp. 115-122.

[4] M. J. Kim, K. J. Kim, S. Kim and A. K. Dey, "Performance Evaluation Gaps in a Real-Time Strategy Game Between Human and Artificial Intelligence Players," in IEEE Access, vol. 6, pp. 13575-13586, 2018.