Iterative Bayesian Search to Determine Opposing Army Location in Starcraft 2

Patrick Hackett
Georgia Institute of Technology
Atlanta, Georgia
patrick.d.hackett@gatech.edu

KEYWORDS

Bayesian Search Theory, Artificial Intelligence, Data Science, Starcraft 2

1 INTRODUCTION

1.1 AI in Professional Gaming

Building AI for video games that can successfully compete with professional players has become a measuring stick for the advancement of AI, with Google's DeepMind recently beating pro Starcraft II players in constrained conditions, and OpenAI recently defeating a professional DOTA 2 team. In this project, I explore the use of Bayesian Search Theory as a potential tool to direct AI search for enemy units across a map in Starcraft 2.[5][1]

Starcraft 2 (SC2) is a highly competitive popular real-time strategy game played across the world. Players choose one out of three races to play as, mine resources to build buildings and armies in real time, and battle over the course of an average of 20 minutes with the goal of destroying an opponent's armies and buildings. [Figure 1]

In Starcraft II, DeepMind applied their AlphaStar algorithm to defeat professional players within a constrained match-up limited to a certain map and race choices.[7] AlphaStar uses a deep neural network trained on thousands of Starcraft II replays equal to 200 years worth of gameplay. The AlphaStar team then split the model into multiple agents at each iteration of training, simulating a human league and taking advantage of population based reinforcement learning. The model uses each frame of the game to predict the remaining sequence of actions using an agent optimized for the particular match-up. In order to determine the next steps in a sequence of actions, the AlphaStar model uses a Bayesian update process to model what is essentially a Markov chain action sequence.

In my model, my goal was to address a particular piece of Starcraft II gameplay by incorporating a Bayesian updating process to guide a theoretical AI in trying to find an opposing army with limited information.

1.2 Bayesian Search Theory

Bayesian search theory was first known to be developed and used in WWII by the US Navy to search for German submarines and, in at least one case, a hydrogen bomb lost at sea. In more recent events, Bayesian search theory has been used to try and locate airplanes lost at sea.[2]

Searches of this nature are often one-time events, where a model cannot be trained thousands of times. Instead, a prior probability distribution is generated that incorporates all known information at that time. Then, the prior distribution is iteratively updated as searches are performed, creating a chain of posterior distributions. Generally, the search space is divided into cells that each have

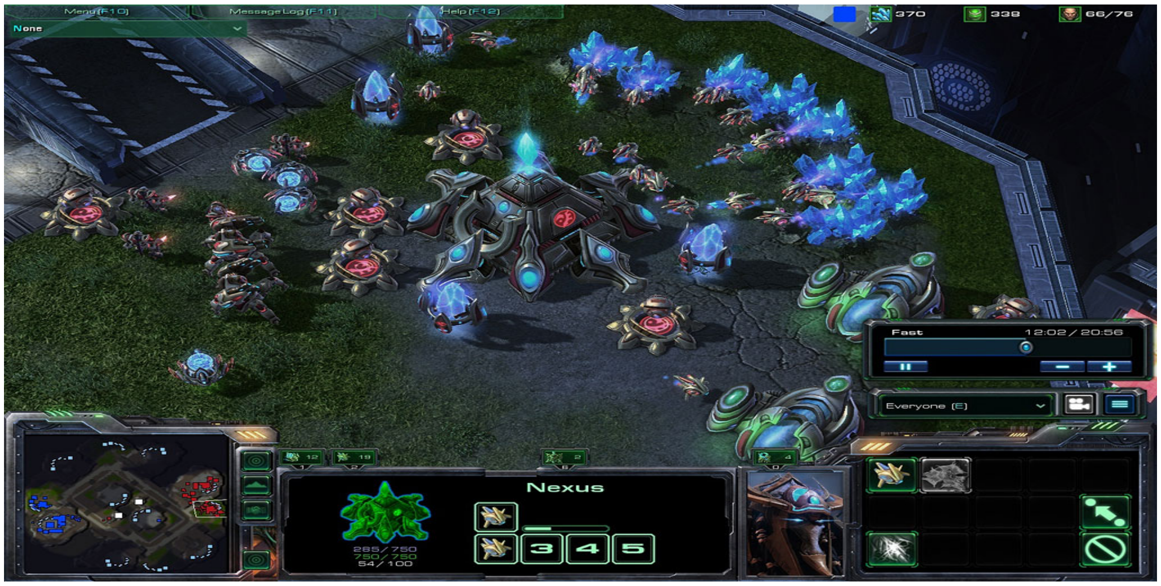

Figure 1: Gameplay photo showing a standard player's base. Includes a primary building called a Nexus, units mining minerals that are used to purchase new buildings and units, as well as a small cluster of army units to the left. The minimap is shown in bottom left.

a probability of discovered the object of interest and search effort is allocated in relation to these probabilities. If the target is not found in the first iteration, the posterior distribution is updated using Bayes' Rule and the search effort is reallocated accordingly. The search also incorporates a chance of failure of finding the object in a grid space where it is present, to account for any number of factors, for example, the difficulty of finding an object in the ocean.

$$\begin{aligned} p(x\ exists\ in\ grid|chance\ of\ finding\ x) \propto \\ p(chance\ of\ finding\ x|x\ exists\ in\ grid) * \\ p(x\ exists\ in\ grid) \end{aligned} \tag{1}$$

Bayesian search theory gives a statistical method for complex searches in situations with limited opportunity for training, allowing for the use of the maximum amount of current information, and providing a framework to determine probability of success.

2 METHODS

2.1 Data

For this model, a professional Starcraft 2 game replay was downloaded, and a time point was chosen during the game where an army was moving across the map. During Starcraft 2 gameplay, a player is only capable of seeing units move in the map where they themselves have a unit present, or are currently "scanning" an area of the map. This is called the "fog of war." Previously explored structures, such as the opponent's base in the top right of Figure 2, are still visible. The image was then coded into a 256 x 256 numpy array in python that marked the location of the army units to be searched for.[Figure 1]

2.2 Prior Probabilistic Map

Based on prior examples, a beta function was determined to be an appropriate conjugate prior for the expected Bernoulli distribution of the data, e.g. whether or not the army exists in a search

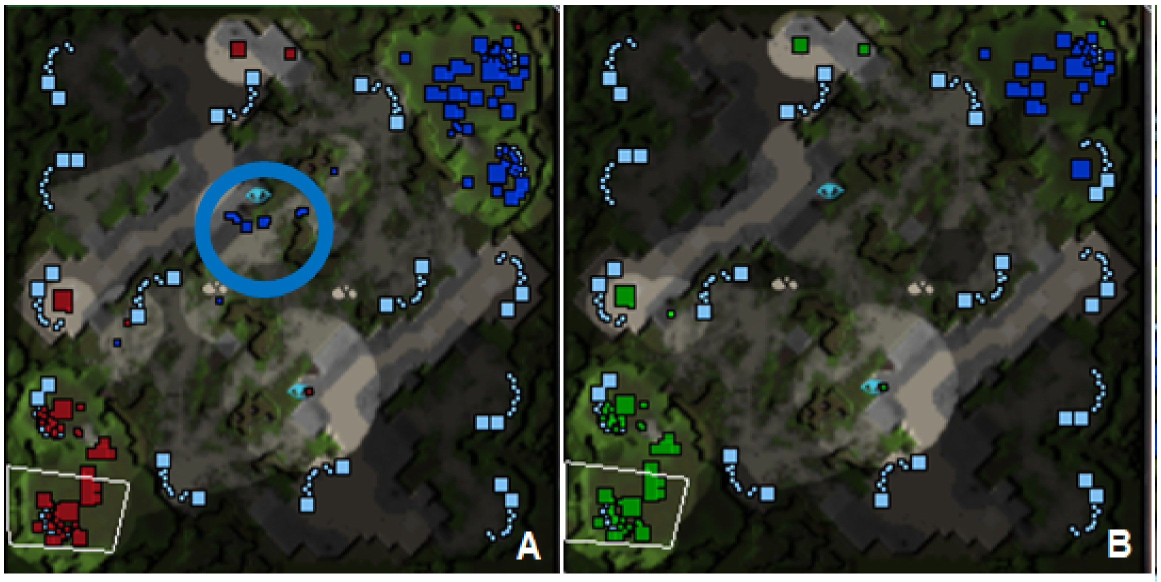

Figure 2: In Image A of the minimap our "hero" is shown in the bottom right, with an opposing army moving across the map (circled in blue). In Image B, we see the map from the perspective of our "hero", where "fog of war" obscures the opposing units.

space.[3, 4, 8]

$$B(\alpha, \beta) = \int_0^1 x^{\alpha-1}(1-x)^{\beta-1}dx$$

A probability distribution was created from the beta function as a function of distance from the enemy base. The army would be most likely to be near the enemy base and the probability would decrease the further out from the source of origin the search takes place. [Figure 3]

The PDF was then flattened and normalized across the map grid to generate a probabilistic search space.[Figure 4]

2.3 Updated Posterior

Once the probabilistic map was built across the search space, the coordinates of the highest probable spot for army location was determined and searched. A circle of 13 px radius was constructed to simulate an in-game "scanner sweep" technology that temporarily reveals the map in the chosen location for 12 seconds.[6]

At that point, if the searched area contains the army, the success is logged and the algorithm stops. If the army is not present within the searched area, the region searched is updated into the probabilistic map, reducing the probability in that region by .99. As a result, the remaining areas are normalized within the new map, and other regions increase the probability of the presence of the army. [Figure 5]

3 RESULTS

From the depiction of the prior probabilistic map, we can see there were flaws in our PDF that limited the algorithm's ability to locate the army.[Figure 4] In the top left of the figure, the distribution created an area of high probability due to the method in which the function was generated solely on a beta distribution across distance from the enemy base . Whereas in the top right we would expect the army to exist close to the enemy base, the top left was out of the path between the two bases. Due

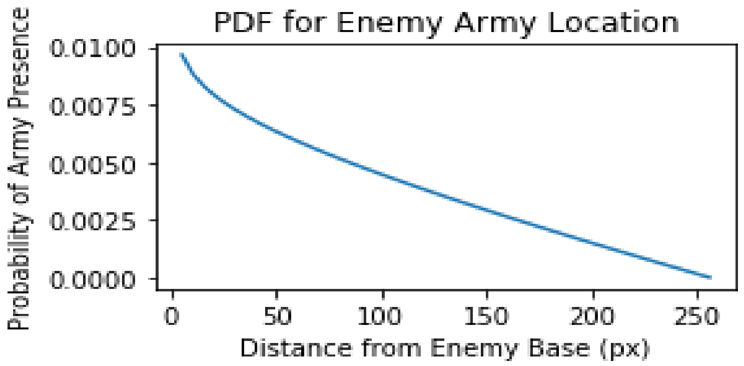

Figure 3: The beta function of probability of army presence as it relates to distance from the enemy base, with higher probability in green.

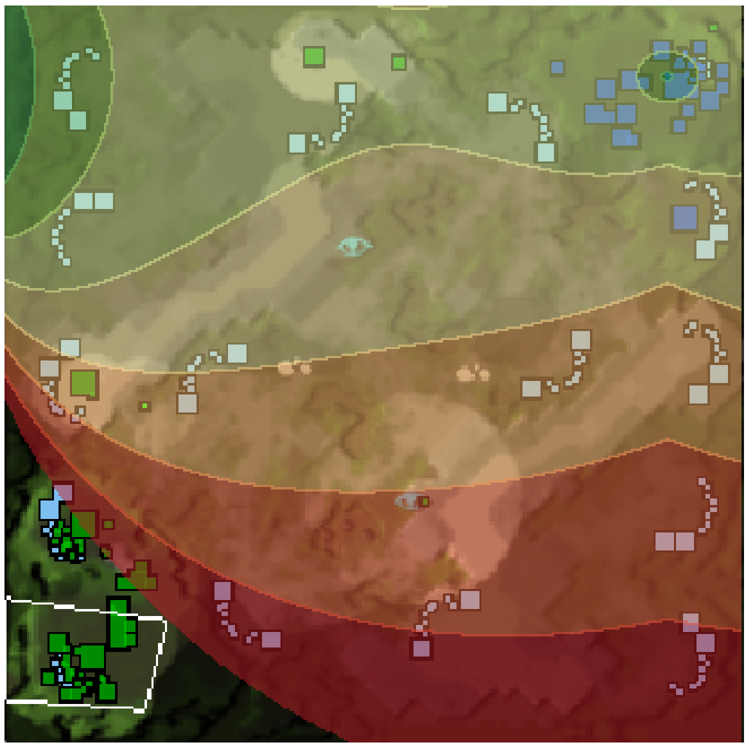

Figure 4: The prior probabilistic map contoured across the minimap plot space

to not manually adjusting the prior, this caused the iterations to continuously increase the remaining high probability spots until the search became stuck in the wrong area. This resulted in the iterative approach failing to find the army.[Figure 6]

4 FUTURE DIRECTIONS AND CONCLUSIONS

4.1 Prior Improvement

While the construction of the updated posterior successfully directed an iterative Bayesian search,

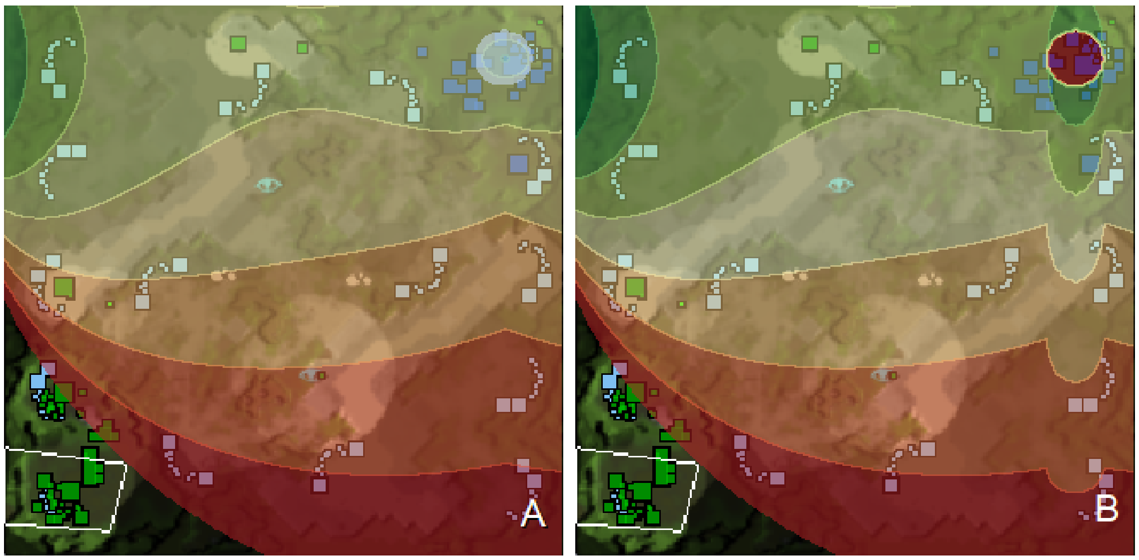

Figure 5: In Image A, the search area is denoted by a white circle at the highest probability area in the prior distribution, directly over the enemy base. In Image B, the map has been updated with the posterior probability distribution as a result of the failed search.

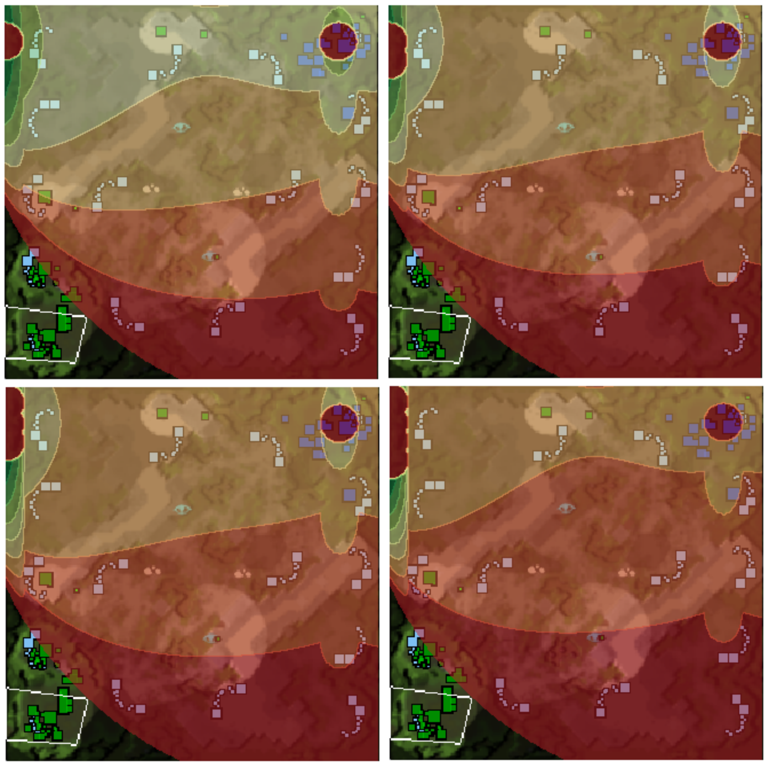

Figure 6: Due to a prior built solely on the conjugate beta function as relating to distance from the enemy base, the iterative search became stuck in the top left quadrant of the map.

due to a lack of prior knowledge, the search failed to discover the enemy army location. The greatest shortcoming of Bayesian search theory is the data originally provided defining the search space.

The initial steps the could be taken to improve this issue is a map terrain analysis and a manual adjustment of the prior taking into consideration the map structure. In this scenario, the prior map would begin with a beta function mapped onto the plot grid, but would be adjusted to lower the probability in the corners, and increase the probability based on terrain "paths" between the two player's bases.

Next steps to improve the ability of this model to detect an army location would to include a trained data set to construct a prior from previous games. For example, by using replays that maintain the same map, race match-up, and starting base locations, a probabilistic map could be generated that would more accurately capture potential army movement at certain time points.

4.2 Posterior Improvement

One of the shortcomings of the posterior update after a search is the lack of incorporation of real-time in-game events. In this constrained scenario, army locations were considered static from search to search. In addition, the scanner sweep search mechanism was considered limitless. In a real-game, the probability of failing to find the army in a location would be much lower than the .01 probability used in our algorithm. This would be due to a constantly moving army. Also, players use multiple methods of searching an area, including sending out their own units as scouts. This would involve a search path rather than a single location, and the incorporation of search failure due to real-time game events and unit movements.

Incorporating these real-time events and various search patterns would greatly increase the complexity of the algorithm beyond the scope of this project, but would have a more direct application for the performance of an AI operating in real-time.

4.3 Conclusions

Bayesian search theory has vast applications in real-world problems. Through this in-game hypothetical application, it is easy to see the power of iterative and automated search based on prior probabilistic maps in guiding the behavior of AI. However, it is clear that this method is limited by the amount of information provided to the algorithm. In taking examples from current AI used in these scenarios, this model could have benefited from a deeper training process to provide a more informative prior distribution.

However, the pace at which AI has succeeded in competing with expert humans in games and other tasks gives evidence to a more thorough training approach having high levels of success. As we continue to move forward in successfully applying AI to more and more problems, we must consider the quality of our information carefully and the data processing steps will continue to grow as one of the most important components of modeling complex algorithms.

It is important to consider the tasks to which we point these machine learning tools and the societal implications. It does not take a stretch of the imagination to consider AI competitive strategy gaming success being extrapolated to actual armies on real battlefields, with very real consequences for human lives. This highlights not only the need for expert analysis for prior determination to guide the algorithms to great accuracy, but also sociological guidance on the part of each data scientist for the goals and outcomes of the algorithms themselves.

REFERENCES

[1] Peter Bright. 2019. OpenAI bot crushes Dota 2 champions, and now anyone can play against it. *Ars Technica* (2019), https://arstechnica.com/gaming/2019/04/openai-bot-crushes-dota-2-champions-and-now-anyone-can-play-against-it/.

[2] J. Van Gurley and Lawrence D. Stone. 2016. Bayesian Search for Missing Aircraft, Ships, and People. *Siam News* (2016), https://sinews.siam.org/Details-Page/bayesian-search-for-missing-aircraft-ships-and-people.

[3] Cameron Mence. 2014. A Bayesian Search for HannibalâĂŹs marauding army during the Second Punic War. *Subsub Routine* (2014), http://www.subsubroutine.com/sub-subroutine/2014/05/18/a-bayesian-search-for-hannibals-marauding-army-during-the-second-punic-war.

[4] Jacob Simmering. 2014. Bayesian Search Models. *R-Bloggers* (2014), https://www.r-bloggers.com/bayesian-search-models/.

[5] Chris Stokel-Walker. 2019. DeepMind AI thrashes human professionals at video game StarCraft II. *NewScientist* (2019), https://www.newscientist.com/article/2191910-deepmind-ai-thrashes-human-professionals-at-video-game-starcraft-ii/.

[6] Liquipedia team. [n. d.]. Scanner Sweep. *Liquipedia* ([n. d.]).

[7] The AlphaStar Team. 2019. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. *DeepMind Technologies Limited* (2019), https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/.

[8] user "gung". 2014. How to apply Bayes' theorem to the search for a fisherman lost at sea. *Stack Exchange* (2014), https://stats.stackexchange.com/questions/119952/how-to-apply-bayes-theorem-to-the-search-for-a-fisherman-lost-at-sea.