Bandit Modeling of Map Selection in Counter-Strike: Global Offensive

Alexander Dong[1] , **Alec Hon**[1] , **Guido Petri**[1] , **Michael Stanley**[1]
[1]New York University
awd275@nyu.edu, abh466@nyu.edu, gp1655@nyu.edu, mhs592@nyu.edu
https://github.com/charlesoblack/csgo-map-bandits

Abstract

Many esports use a pick and ban process to define the parameters of a match before it starts. In Counter-Strike: Global Offensive (CSGO) matches, two teams first pick and ban maps, or virtual worlds, to play. Teams typically ban and pick maps based on a variety of factors, such as banning maps which they do not practice, or choosing maps based on the team's recent performance. We introduce a contextual bandit framework to tackle the problem of map selection in CSGO and to investigate teams' pick and ban decision-making. Using a data set of over 3,500 CSGO matches and over 25,000 map selection decisions, we consider different framings for the problem, different contexts, and different reward metrics. We find that teams have suboptimal map choice policies with respect to both picking and banning. We also define an approach for rewarding bans, which has not been explored in the bandit setting, and find that incorporating ban rewards improves model performance. Finally, we determine that usage of our model could improve teams' predicted map win probability by up to 11% and raise overall match win probabilities by 19.8% for evenly-matched teams.

1 Introduction

As data acquisition methods become more pervasive, sports analytics has received increased interest in contemporary sports, like soccer, basketball and baseball [Assunção and Pelechrinis, 2018]. One common application in sports analytics is valuing player actions and decision-making. For example, Decroos et al. introduce a framework to value soccer players according to how their actions change their team's chance of scoring [Decroos *et al.*, 2019].

Esports, also known as professional video gaming, is one of the fastest growing sports markets in the world. Yet esports has attracted little sports analytics interest. Most analytical work in esports covers massively online battle arena (MOBA) games, such as League of Legends or Defense of the Ancients 2 ("DOTA2"). Accordingly, there exists a dearth of work on Counter-Strike: Global Offensive (CSGO), one of the oldest yet most popular esports. A picking and banning process is a common process in many esports, where some entities are banned from being played in a particular game. For example, in League of Legends, teams ban a set of characters, and their players each pick a character to play before the game starts. In CSGO, teams typically perform a map selection process where each team takes turns picking and banning maps to play. However, map selection routines are often not based on analytics and data, but rather on players' inclinations at selection time.

Contextual bandits are statistical models that learn a conditional distribution (a "policy") over an action space conditioned on a context and a reward which is itself conditioned on both the context and action. The bandit learns a policy with the objective of maximizing the reward returned by the action taken. Significantly, no rewards are known for actions not taken. In this paper, we apply a contextual bandit framework to the domain of map selection in CSGO. We use a novel data set of over 25,000 map pick and ban decisions from over 3,500 professional CSGO matches to train three different bandit framings to the problem. We find that teams' choices in the map selection process are suboptimal and do not yield the highest expected win probability.

The paper is structured accordingly. In section 2, we review relevant esports and contextual bandit works. In section 3, we cover CSGO's map selection process. In section 4, we introduce our contextual bandit model. In section 5, we describe our dataset and our evaluation methodology. Section 6 contains our results. We discuss the benefits of our model, the choices of evaluation metrics and suggest possible areas of future work in section 7 and conclude the paper in section 8.

2 Related Work

Reinforcement learning (RL) techniques are increasingly being applied to sports analytics problems. Liu et al. first used RL in sports to estimate an action-value Q function from millions of NHL plays [Liu and Schulte, 2018]. They used the learned Q function to value players based on the aggregate value of their actions. Liu et al. also apply mimic learning to make their models more interpretable [Liu *et al.*, 2018]. Sun et al. extend this work by considering a linear model tree [Sun *et al.*, 2020]. While the previous works heavily focused on ice hockey, Liu et al. also learn an action-value

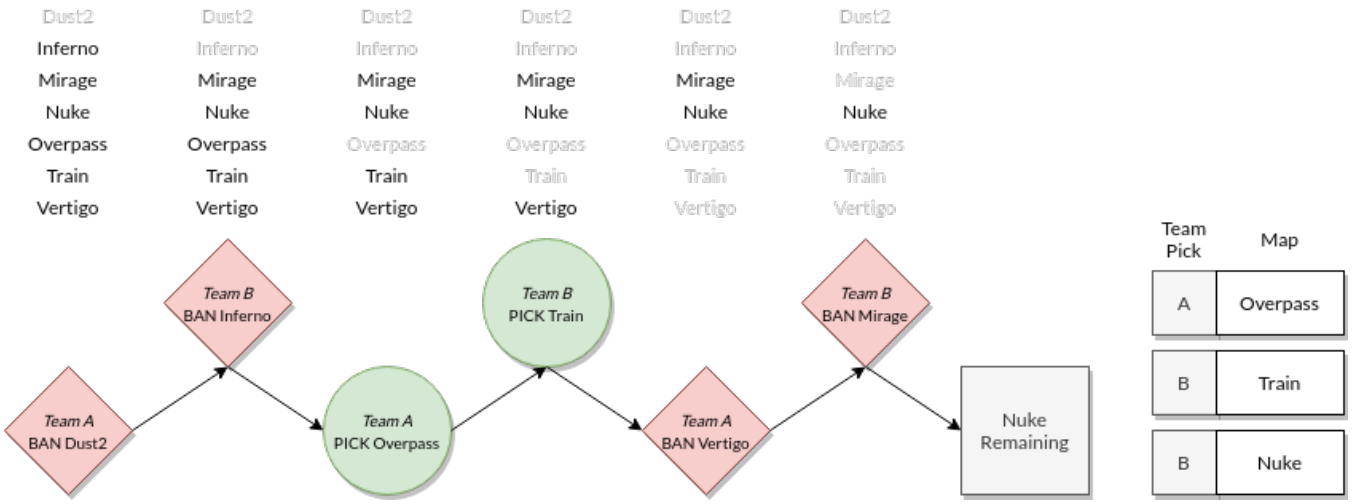

Figure 1: Example map selection process for a best-of-three match. The available map pool is shown above each pick/ban decision. The first team, usually decided by tournament rules, bans a map. The second team then does the same. The two teams then both pick a map, and then both ban a map. In total, there are six decisions, four of which are bans, and two are picks.

Q function for soccer [Liu *et al.*, 2020]. Despite the heavy use of other RL approaches such as Q-learning, contextual bandits have not been as heavily utilized in sports analytics.

This paper applies contextual bandits to the multi-arm map selection process in esports matches for the game CSGO. Contextual bandits are a simplified case of reinforcement learning. In reinforcement learning, an action is chosen based on the context (or state) and a reward is observed, and this process is repeated for many rounds. Rewards are not observed for actions not chosen. In the contextual bandit case, the contexts of different rounds are independent. [Tewari and Murphy, 2017] provides a thorough review of contextual bandits, tracing the concept back to [Woodroofe, 1979] and the term back to [Langford and Zhang, 2008]. Many approaches have been explored for learning policies in the contextual bandit setting. [Williams, 1992] introduced gradient approaches in the reinforcement learning setting, and [Sutton and Barto, 2018] applied the approach to the specific case of contextual bandits. Comparing proposed policies often requires off-policy evaluation: estimating the value of a policy from data that was generated by a different policy (the "logging policy"). This paper utilizes two off-policy evaluation approaches: the self-normalized importance-weighted estimator [Swaminathan and Joachims, 2015] and the direct method of regression imputation [Dudík *et al.*, 2014]. To our knowledge, ban actions have never been modeled in the bandit setting.

Esports have mostly attracted sports analytics interest in the form of win prediction and player valuation. Numerous efforts have been made to predict win probabilities in popular esports games such as CSGO and DOTA2. [Yang *et al.*, 2016] and [Hodge *et al.*, 2019] first use logistic regression and ensemble methods to predict win probabilities in DOTA2, a popular MOBA game. [Makarov *et al.*, 2017] first predicted CSGO win probabilities using logistic regression, however their data only included less than 200 games. [Xenopoulos *et al.*, 2020] expand on previous CSGO work by introducing a data parser and an XGBoost based win probability model for CSGO. They also value players based on how their actions change their team's chance of winning a round. [Bednárek *et al.*, 2017] value players by clustering death locations.

Map selection is a process largely unique to CSGO and has not been well studied, but is loosely related to another esports process unique to MOBA games: hero selection. In DOTA2, for example, players from opposing teams alternate choosing from over one hundred heroes, with full knowledge of previous hero selections. [Yang *et al.*, 2016] and [Song *et al.*, 2015] use the selected heroes as features to predict win probability, but do not recommend hero selections or explicitly model the selection process. More relevant is the hero selection recommendation engine of [Conly and Perry, 2017], which uses logistic regression and K-nearest neighbors to rank available heroes based on estimated win probability; they do not, however, consider historical team or player context.

3 Counter-Strike Map Selection

Counter-Strike is a popular esport that first came out in 2000, and CSGO is the latest version. The game mechanics have largely stayed the same since the first version of the game. Before a CSGO match starts, two teams go through the map selection process to decide which maps the teams will play for that match. A map is a virtual world where CSGO takes place. Typically, matches are structured as a best-of-three, meaning the team that wins two out of three maps wins the match. A team wins a map by winning rounds, which are won by completing objectives. In each round, one team plays as the *terrorist team* (T), and another team plays as the *counter-terrorist team* (CT). The teams play 15 rounds as their respective side, and switch on the 16th round.

The collection of available maps in the map selection process is called the *map pool*. Typically, there are seven maps

to choose from in the map pool. Although the maps rarely change, a new map may be introduced and replace an existing map. Our data contains map selections using the following map pool: `dust2`, `train`, `mirage`, `inferno`, `nuke`, `overpass`, `vertigo`. The map selection process is exemplified in Figure 1. First, team A *bans* a map. This means that the teams will not play the map in the match. The team that goes first in the map selection process is usually higher seeded, or determined through tournament rules. Next, team B will ban a map. The teams then will each *pick* a map that they wish to play in the match. Next, team A will ban one more map. At this point, team B will ban one of the two remaining maps, and the map not yet picked or banned is called the *decider*.

Professional teams may sometimes have what is referred to as a *permaban* – a map that they will always ban with their first ban. For example, some teams may choose to ban the same map in over 75% of their matches. From interviews with four CSGO teams ranked in the top 30, two of which are in the top 10, teams choose their maps from a variety of factors. Some teams try to choose maps they have a historically high win percentage, or maps where their opponents have low win percentages. Other teams may also choose maps purely based on how their recent practice matches performances.

4 Bandit Model for CSGO Map Selection

In order to model the map selection process, we elected to use a *k*-armed contextual bandit. This was a clear choice: the actions taken by teams only yield a single shared reality, where we cannot observe the counterfactual of different choices. The bandit model enables us to approximate the counterfactual reality and frame this problem as a counterfactual learning problem.

In particular, we used the context from teams' previous matches, as well as information available at the time of selection, such as which maps were still in the selection pool. There are two kinds of actions: picks and bans, which must be manipulated differently. The reward is the map being won by the choosing team or not, as well as more granular version of this in which we include margin of victory.

4.1 Context and Actions

Our initial choice for the context given a particular round t in the map-picking process was a one-hot encoding for the available maps in that particular round, such that the bandit would learn to not pick the map if it was not available. To give the bandit more information about the teams that were deciding for that particular match, we implemented two historical win percentages, the first being the team's historical match win percentage, and the second being the team's historical map win percentage for each map. The first percentage is utilized to indicate team strength compared to other teams, and the second the team's overall ability to play well on each map. We applied Laplace smoothing to the initial percentages for numerical stability, using the formula

$$\text{Win\%} = \frac{\text{Wins} + 5}{\text{Matches} + 10}. \tag{1}$$

Both win percentages were stored in the context vector for both the deciding team and the opponent team alongside the available maps. For both picks and bans, the given *context* is the same as described above, and the corresponding *action* would be the map picked or banned by the deciding team.

4.2 Rewards

Picks

Due to the nature of the map-picking process, where the decider is a forced pick, we chose to remove the rewards from all final map picks, as it would not make sense to reward either team for a forced choice. As a result, only the first two picks from each map selection process were given a reward. Rewards for map-picking were implemented with two different methods. Our first method utilized a simple 0-1 reward ("0/1"), where if the deciding team won on the map they had picked, they would be rewarded with an overall reward of 1 for that action, or 0 otherwise. Our second method rewarded the deciding team based on the margin of rounds won ("MoR") in the best-of-30 rounds on the decided map. The reward function for deciding team i and an opponent team j is given below:

$$R_{i,j} = \frac{\text{Rounds won by } i - \text{Rounds won by } j}{\text{Total number of Rounds on map}} \tag{2}$$

The round proportion rewards were implemented as a more granular method to compare team performance on each map.

Bans

Since there is no data on how any deciding team would perform on a banned map, we chose to reward bans based on the deciding team's overall performance in the match, where if the deciding team won the match, they would be rewarded for choosing to not play on the banned map with an overall reward of 1, or, if they lost, a reward of -1. In addition, we implemented a exponentially decreasing reward over the ban process, where earlier bans would have higher rewards. Later map picks have fewer available choices: restricting the action space means a team may be forced to make a choice they do not want, and so we de-emphasize the later choices. The ban reward function for team i playing in match t is given below:

$$R_{i,t}(n) = \begin{cases} 1 \cdot \frac{1}{2^n} & \text{if team } i \text{ won match } t \\ -1 \cdot \frac{1}{2^n} & \text{if team } i \text{ lost match } t \end{cases} \tag{3}$$

where n is the nth ban in the map picking process. In our case, $n \in \{1, 2, 3, 4\}$, as there are always four bans in the map picking process for CSGO.

4.3 Policy Gradient Learning

The most straightforward way to train a bandit is via policy gradient learning [Sutton and Barto, 2018]. For our policy class, we use a multinomial logistic regression parameterized by weights θ and an action-context mapping function $\phi(x, a)$, with the softmax function to transform the affinity of the bandit for each action into a probability:

$$\pi(a|x) = \frac{\exp(\theta^T \phi(x, a))}{\sum_{i=1}^{k} \exp(\theta^T \phi(x, i))} \tag{4}$$

The policy gradient approach trains the model via SGD [Sutton and Barto, 2018], enabling both online and episodic learning. In particular, the optimization maximizes the expected reward for the bandit, using the update function

$$\theta_{t+1} \leftarrow \theta + \eta R_t(A_t) \nabla_\theta \log \pi_{\theta_t}(A_t|X_t) \tag{5}$$

with π defined above and the gradient

$$\nabla_\theta \log \pi(a|x) = \phi(x,a) - \frac{\sum_{i=1}^{k} \phi(x,i) \exp(\theta^T \phi(x,i))}{\sum_{i=1}^{k} \exp(\theta^T \phi(x,i))}. \tag{6}$$

In the context of picks, we can use online learning to iteratively update the parameters θ. For bans, however, we do not observe a reward at the time the choices are made; as a result, we used episodic learning, where an episode is an entire match.

5 Experiments

5.1 Data

We obtained our data from HLTV.org, a popular CSGO fan site. The site contains scores, statistics and map selections for most professional matches. In particular, we had access to information about which teams played, the map selection record, and the per-round winning team. Additionally, we had information on when each match was played and which tournament it was a part of. We do not use any in-game replay features in our modeling.

We used matches from April 2020 to March 2021. In total, this consisted 628 teams that played a total of 6283 matches, summing to 13154 games. We only consider best-of-three matches, which are by far the most popular match format. We focus on the games where the most common set of seven maps is selected from the map pool of `dust2`, `inferno`, `mirage`, `nuke`, `overpass`, `train`, `vertigo`. In addition, we also remove teams such that in the final dataset, each team has played at least 25 games, or approximately 10 matches, with another team in the dataset. This leaves us with 165 teams, playing a total of 3595 matches, summing to 8753 games. The resulting dataset was split into an 80-20 train-test split by matches for bandit learning and evaluation.

Repeated Matchups

We made the assumption that the logging policy is static. In reality, we would expect that if two teams consistently played against another, then they would start to develop a metagame between them. In figure 2, we show a histogram of repeated team matchups. We can see that the majority of matches are first or second time occurrences, and rarely do teams play each other more than five times in a formal, competitive setting. In addition, the sparsity of repeated matchups influenced the modeling decision to not use an indicator variable for each specific team, which is elaborated in section 5.3.

Round Differentials

A map consists of 30 rounds, where the winner must win 16 of them. Should the score be tied, the teams go into *overtime*, where teams will play six further rounds at a time until a win by two is achieved. In figure 3, we present a graph showing

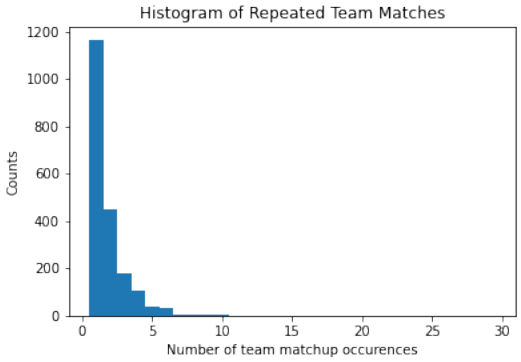

Figure 2: Histogram of the number of times a given pair of teams played each other. There are very few teams that played each other several times, so we assume there is no specific adaptation of team strategy for given opponents.

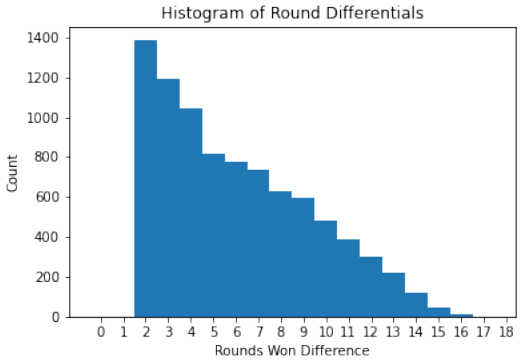

Figure 3: Incidence of round win differentials between winning and losing team at match end. There is a strong decay factor indicating that most teams are evenly matched.

a histogram of round differentials at the end of each game. As noted by the histogram, a large number of matches were won with round differentials less than 4, and there appears to be a linear decay in counts with increasing round differentials wins. This implies that most teams playing against each other are evenly matched, and that there are very few cases were a team is completely outmatched that they would lose a large majority of the matches. We chose to use the round differentials as a reward because it would provide us with not only whether the deciding team won the match, but would also tell us how well the team performed against its opponent.

Map Win Rates

In table 1, we present the mean and variances of win percentages for teams on each map. The relatively large variances indicate that teams have varying win rates across maps. We can conclude there is opportunity for an optimal pick and ban policy to create value by selecting maps where one of the teams has an advantage or disadvantage.

Feature Engineering

As described in section 4.1, our basic feature engineering consisted of creating one-hot encodings for the available

Map	T Win Var	CT Win Var	T Win Mean	CT Win Mean
dust2	6.3%	5.7%	52.5%	48.4%
inferno	4.7%	5.2%	52.1%	48.9%
mirage	5.7%	5.4%	48.6%	52.4%
nuke	5.5%	9.0%	46.8%	54.3%
overpass	5.7%	6.9%	48.9%	52.2%
train	4.8%	7.4%	46.3%	55.6%
vertigo	11.0%	11.6%	52.4%	48.1%

Table 1: Win percentage mean and variances for each map. The high variances indicate that there is a substantial difference between different teams playing on the same map, which implies there may be a big win probability advantage to selecting maps correctly.

maps in each map-picking round. Our advanced feature engineering consisted of the picking team and opposing team's historical match win percentage as well as map win percentages with added Laplace Smoothing for numerical stability, We used historical win percentages to avoid leakage of data from the current match to influence the bandit. Rewards were engineered following the process described in section 4.2.

5.2 Evaluation

We use two typical off-policy evaluation methods, the *direct method* ("DM") [Dudík *et al.*, 2014] and the self-normalized importance-weighted estimator ("SN-IW") [Swaminathan and Joachims, 2015]. We also present the mean reward observed as a baseline.

The goal of the direct method is to estimate the reward function $r(x, a)$ that returns the reward for any given action a for the context x. We estimate the reward function by using an importance-weighted ridge regression for each action. We use the self-normalized importance-weighted estimator with no modifications.

Value estimates are presented for four different reward settings:

- Picks(0/1): Expected pick reward for models trained with 0/1 rewards
- Picks(MoR): Expected pick reward for models trained with MoR rewards
- Bans(0/1): Expected ban reward for a models trained with 0/1 rewards
- Bans(MoR): Expected ban reward for models trained with MoR rewards

5.3 Baseline

For our initial policies, we utilized the initial `Logging Policy` and a `Uniform Policy`. The `Uniform Policy` is a simple uniform distribution over the k possible bandit arms. The true `Logging Policy` is not known, as teams do not share their map selection approaches. We estimated the logging policy as the empirical action distribution for each team. Conditioning the logging policy on additional context would be desirable but would also require more data than is available.

5.4 Variety of Policies

We experimented with three different varieties of contextual bandits: `SplitBandit`, `ComboBandit`, and `EpisodicBandit`.

`SplitBandit` is composed of two individual, simple contextual bandits, each with a θ parameter size of $(\texttt{n\_features} \cdot \texttt{n\_arms})$. The first contextual bandit is trained on the picks via online learning. The second contextual bandit is trained on the bans in an episodic fashion. We also built a version of `SplitBandit` that used a θ parameter of size $(2 \cdot \texttt{n\_features} \cdot \texttt{n\_arms})$, containing each individual bandit.

`ComboBandit` is a single model also trained on the picks via online learning and on the bans via episodic learning with a θ parameter size of $(\texttt{n\_features} \cdot \texttt{n\_arms})$. `ComboBandit` learns a single set of parameters that define a policy for both picks and bans. The parameters directly define a pick policy by Eq(4) and the ban policy is derived from that pick policy:

$$\pi_B(a|X) = \frac{1 - \pi_P(a|X)}{\sum_{\alpha \in A} 1 - \pi_P(\alpha|X)} \tag{7}$$

for pick policy π_P and ban policy π_B over actions A and context X.

`EpisodicBandit` is similarly a single model, but it is trained on both the picks and bans simultaneously via episodic learning with a θ parameter size of $(2 \cdot \texttt{n\_features} \cdot \texttt{n\_arms})$. We expected this model to perform similarly to `SplitBandit`, since its gradient estimates are better estimates than the estimates derived from individual datapoints, offsetting the quicker adaptability of the online gradient calculation with less noise.

6 Results

Our main results are summarized in table 2. We discarded the joint version of `SplitBandit` owing to its lack of theoretical backing and its poor performance: training each section separately induced a form of regularization by driving the individual parameters of θ to the origin, leading to a worse performance than the separate models.

Considering the self-normalized estimator, the best model for picks was `SplitBandit` trained on proportional rewards, while the best model for bans was `ComboBandit` trained on proportional rewards. The uniform policy performs better than the logging policy for the picks in our

	Picks (0/1)	Picks (MoR)	Bans (0/1)	Bans (MoR)
Uniform policy (split)	0.568/0.541	0.568/0.541	-0.018/-0.003	-0.018/-0.003
Logging policy	0.549/0.549	0.549/0.549	-0.014/-0.014	-0.014/-0.014
SplitBandit	0.587/0.554	**0.659/0.528**	-0.016/0.004	-0.016/0.004
ComboBandit	**0.640/0.528**	0.613/0.573	**0.021/0.003**	**0.036/-0.015**
EpisodicBandit	0.568/0.551	0.561/0.547	0.013/0.006	0.013/0.006

Table 2: Expected reward for each policy type under four different evaluations. The best policy parameters were found via grid search and the policy was optimized with policy gradient. Both the SN-IW (left) and DM (right) evaluation methods are presented, except for Logging policy where the on-policy value is presented. Every model tested outperforms or matches the baseline uniform policy, with the best overall model being the bandit trained on both picks and bans. Comparisons between the uniform and logging policy indicate teams choose their bans well, but their picks poorly.

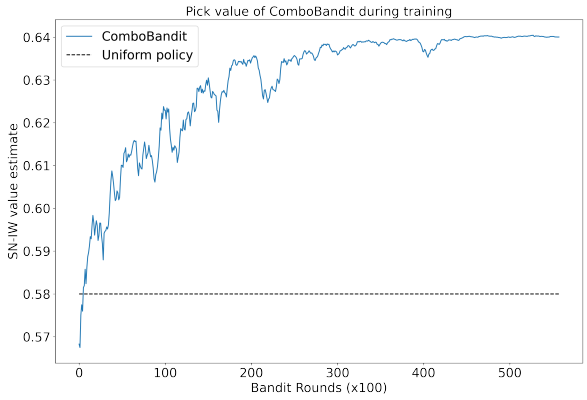

Figure 4: Picks(0/1) value on the test set for `ComboBandit` and Uniform policy, evaluated every 100 rounds over 3 epochs of training. The bandit quickly surpasses the uniform policy's performance and plateaus around an expected reward value of approximately 0.64.

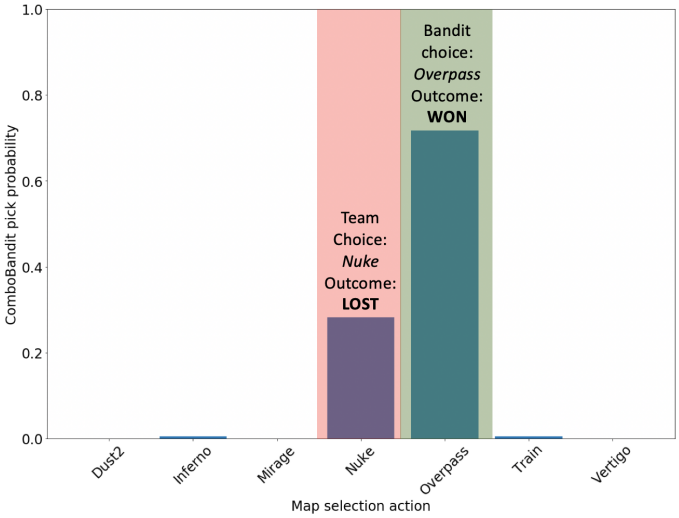

Figure 5: The best model's probability distribution for pick 4 in a match between *TIGER* and *Beyond*. *TIGER*, the deciding team, chose `Nuke` and lost the map, later going on to win map `Overpass`, which was `ComboBandit`'s suggestion.

dataset but worse for bans, which indicates teams' picks might be overconfident, whereas their bans are chosen more carefully.

`ComboBandit` substantially outperforms all other policies. We believe this is due to its training including the additional data from both picks and bans instead of selecting only one of the two categories for training a given parameter. This yields a better optimization through better gradient estimates. `EpisodicBandit` is trained on both picks and bans, but its parameters do not depend on both subsets of data, which does not provide that optimization advantage. The learning curve in Figure 4 shows that ComboBandit surpasses the uniform policy benchmark after only a few training rounds, continuing to improve over 3 epochs of training.

Figure 5 shows an example of `ComboBandit`'s policy. In this match, team *TIGER* chose to play on the map `Nuke`, which they later lost. `ComboBandit` suggested instead to play on `Overpass`, with 71% probability. In the same match, `Overpass` was chosen as the decider and *TIGER* won that map, indicating that the bandit model's policy distribution was more valuable than the team's intuition on map choice.

7 Discussion

The results indicate that teams using our chosen policy instead of their traditional map-picking process can increase their expected win probability by 9 to 11 percentage points, depending on the policy used. This is a substantial advantage for a best-of-3 match, since the model could confer that added win probability to all three map choices. Similarly, the ban choice can be improved by using our model. The logging policy yields an expected reward of approximately -0.014, which indicates that bans have a slight negative effect on match win probability. However, our best model's expected reward for bans is 0.036, thus increasing match win probability by approximately 5% after a ban choice. For two teams that are evenly matched, using our bandit for both pick and ban decisions translates to the team that uses the model having an expected overall match win probability of 69.8% instead of 50%, a substantial advantage for a team.

The choice of evaluation metric is particularly important in examining the results. Using the direct method instead of the self-normalized estimator, we reach drastically different

conclusions about which model to use, with the best overall model being `EpisodicBandit`. In our experiments, we used ridge regressions for our regression imputation. This is clearly a suboptimal model for this estimation, since the context features of win probabilities are bounded: there is a non-linear relationship between the context and the rewards. This is a big limitation of our experiments: we instead relied on the importance-weighted estimator, which is known to be imprecise in estimating policies far from the logging policy.

Future work in this area will be concentrated on examining better choices for evaluation metrics, as well as expanding the contextual features further by adding, for example, player turnover, team-based Elo metrics or rankings, or examining recent performances, such as win percentage in the last 10 matches. The rewards can also be expanded by using not only margin of rounds won per map, but also the margin of players alive per map at the end of a round. Additionally, different framings for the bandit can be considered, such as creating a ranking of which maps are best to choose instead of the model selecting a single map for the user. Our approach can also be extended to other similar processes in esports, such as hero choices in MOBA games.

8 Conclusion

We modeled the map selection process in Counter-Strike: Global Offensive as a bandit, framing the problem in several different ways. Our key contributions are (1) the introduction of bandits and simple reinforcement learning models to esports and CSGO in particular, and (2) novel ways of implementing negative choices in bandits, for which we explicitly choose not to observe their rewards. We find that our model shows that teams are making sub-optimal map selections.

Acknowledgments

Peter Xenopoulos, David Rosenberg

References

[Assunção and Pelechrinis, 2018] Renato M. Assunção and Konstantinos Pelechrinis. Sports analytics in the era of big data: Moving toward the next frontier. *Big Data*, 6(4):237–238, 2018.

[Bednárek *et al.*, 2017] David Bednárek, Martin Kruliš, Jakub, Yaghob, and Filip Zavoral. Player performance evaluation in team-based first-person shooter esport. *International Conference on Data Management Technologies and Applications*, pages 154–175, July 2017.

[Conly and Perry, 2017] Kevin Conly and Daniel Perry. How does he saw me? A recommendation engine for picking heroes in Dota 2. Unpublished course project, 2017.

[Decroos *et al.*, 2019] Tom Decroos, Lotte Bransen, Jan Van Haaren, and Jesse Davis. Actions speak louder than goals: Valuing player actions in soccer. In Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis, editors, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 1851–1861. ACM, 2019.

[Dudík *et al.*, 2014] Miroslav Dudík, Dumitru Erhan, John Langford, and Lihong Li. Doubly robust policy evaluation and optimization. *Statistical Science*, 29(4), Nov 2014.

[Hodge *et al.*, 2019] Victoria J. Hodge, Sam Devlin, Nick Sephton, Florian Block, and Peter I. Cowling. Win prediction in multi-player esports: Live professional match prediction. *IEEE Transactions on Games*, 2019.

[Langford and Zhang, 2008] John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. *In Advances in neural information processing systems*, pages 817–824, 2008.

[Liu and Schulte, 2018] Guiliang Liu and Oliver Schulte. Deep reinforcement learning in ice hockey for context-aware player evaluation. In Jérôme Lang, editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 3442–3448. ijcai.org, 2018.

[Liu *et al.*, 2018] Guiliang Liu, Wang Zhu, and Oliver Schulte. Interpreting deep sports analytics: Valuing actions and players in the NHL. In *Proceedings of the 5th Workshop on Machine Learning and Data Mining for Sports Analytics co-located with 2018 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2018), Dublin, Ireland, September 10th, 2018*, volume 2284 of *CEUR Workshop Proceedings*, pages 69–81. CEUR-WS.org, 2018.

[Liu *et al.*, 2020] Guiliang Liu, Yudong Luo, Oliver Schulte, and Tarak Kharrat. Deep soccer analytics: learning an action-value function for evaluating soccer players. *Data Min. Knowl. Discov.*, 34(5):1531–1559, 2020.

[Makarov *et al.*, 2017] Ilya Makarov, Dmitry Savostyanov, Boris Litvyakov, and Dmitry I. Ignatov. Predicting winning team and probabilistic ratings in "Dota 2" and "Counter-Strike: Global Offensive" video games. *International Conference on Analysis of Images, Social Networks and Texts*, pages 183–196, July 2017.

[Song *et al.*, 2015] Kuangyan Song, Tianyi Zhang, and Chao Ma. Predicting the winning side of Dota 2. Unpublished course project, 2015.

[Sun *et al.*, 2020] Xiangyu Sun, Jack Davis, Oliver Schulte, and Guiliang Liu. Cracking the black box: Distilling deep sports analytics. In Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash, editors, *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 3154–3162. ACM, 2020.

[Sutton and Barto, 2018] Richard Sutton and Andrew Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 2018.

[Swaminathan and Joachims, 2015] Adith Swaminathan and Thorsten Joachims. The self-normalized estimator for counterfactual learning. *In Advances in neural information processing systems*, pages 3231–3239, 2015.

[Tewari and Murphy, 2017] Ambuj Tewari and Susan A. Murphy. From ads to interventions: Contextual bandits in mobile health. In *Mobile Health*, pages 495–517. Springer, Cham, 2017.

[Williams, 1992] Ronald Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.

[Woodroofe, 1979] Michael Woodroofe. A one-armed bandit problem with a concomitant variable. *Journal of the American Statistical Association*, 74(368):799–806, 1979.

[Xenopoulos *et al.*, 2020] Peter Xenopoulos, Harish Doraiswamy, and Cláudio T. Silva. Valuing player actions in Counter-Strike: Global Offensive. In *IEEE International Conference on Big Data, Big Data 2020, Atlanta, GA, USA, December 10-13, 2020*, pages 1283–1292. IEEE, 2020.

[Yang *et al.*, 2016] Yifan Yang, Tian Qin, and Yu-Heng Lei. Real-time esports match result prediction. *arXiv preprint*, arXiv:1701.03162, 2016.