EsportsBench: A Collection of Datasets for Benchmarking Rating Systems in Esports

Clayton Thorrez
claytonthorrez@gmail.com

Abstract

Rating systems are widely used for ranking players and teams, predicting match outcomes, and matchmaking in various sports and games. However, new rating system methods are often evaluated using only a few datasets, with variations in the selected sports/games, leagues, and date ranges, leading to inconsistent benchmarks that hinder comparison across research efforts. Esports, the highest level of video game competition, has seen rapid growth over the past decade, attracting large audiences and significant cash prizes. The extensive match data available on fan sites and wikis makes esports data a valuable resource for rating systems research. We introduce EsportsBench, a curated collection of esports datasets spanning over 20 years and encompassing a diverse range of game genres and competition formats. We conduct experiments benchmarking the predictive performance of various rating systems and recommend methodologies for future ratings systems research on esports data. EsportsBench is available at https://hf.co/datasets/EsportsBench/EsportsBench.

1 Introduction

In competitive games and sports, rating systems assign numerical values, to competitors representing their underlying skill. These systems are fundamental for popular applications including prediction, ranking, and matchmaking. Predicting the results of future matches is a common practice among professional analysts, odds-setters, and fans. Ranked lists of competitors are often constructed either as a part of an official league format, or by media and fans for entertainment. Ratings are also used to ensure balanced match-ups either as seeding mechanisms, qualification criteria, or in the case of video games, as a component of skill based match-making systems. All of these applications rely on the underlying rating system providing accurate approximations of the true strengths of competitors, motivating extensive research into the development of accurate and efficient rating systems.

Effective rating systems research involves running carefully designed experiments on datasets of competitive matches. To facilitate the evaluation and development of rating systems, we introduce EsportsBench, a suite of large and diverse datasets comprised of competitive esports data. To establish a baseline for future comparisons, we conduct experiments evaluating several existing rating systems using these new datasets and share our findings to guide future research.

2 Background

Much of the work in the field of rating systems stems from research centered around the rating of chess players (Elo, 1961; 1967; 1978; Glickman, 1995a; Glickman and Jones, 1999). As the field developed and rating systems were recognized as effective methods for ranking, prediction, and matchmaking, they were applied to a wide variety of other games and sports including tennis (Glickman, 1995b; Ingram, 2021; Hua et al., 2023), American football (Glickman and Stern, 1998; Glickman, 2001) Soccer, (Fahrmeir and Tutz, 1994), hockey (Szczecinski and Tihon, 2023), and

Preprint. Under review.

volleyball (Glickman et al., 2018). Several works have also included comparisons on data from video games such as online Halo matches (Herbrich et al., 2006; Weng and Lin, 2011).

While it speaks to the efficacy of rating systems that they are widely used across different competitions, the wide range of competition datasets has the unfortunate result that many methods are not compared to each other on the same sports or games, and even when results on the same sport are reported, they are often for different time ranges or leagues. This makes it difficult to compare rating systems, as different experiments may reach opposing conclusions about the predictive performance of rating systems when those differences may be attributable to the properties of their datasets.

Inspired by the widespread use of standardized datasets for model comparisons in other machine learning subfields such as computer vision and natural language processing, we curate and process esports match datasets in standardized formats. We then conduct baseline experiments using a wide range of rating systems in order to provide a jumping off point for future rating systems research. The EsportsBench datasets span broad time ranges with different distributions over those ranges and differ substantially in size, outcome variance, competitor distribution, competition format, and gameplay mechanics. We hope that the diversity of the datasets can help provide clarity into the situations in which different rating systems perform optimally and inspire new methods informed by that knowledge.

2.1 Notation and Setup

We define a match m as a tuple representing the meeting of two competitors, c_a and c_b at time t and having outcome y. The competitor identifiers c_a and c_b are integers in the set $\{1...C\}$ with C being the total number of competitors. The outcome $y \in \{1, 0.5, 0\}$ is defined with 1 representing a win for c_a, 0.5 a draw and 0 a loss. We use $t \in \{1...T\}$ to represent the index of a rating period. A rating period is a predefined length of time such as a day, week, month, or year, where all matches are considered simultaneous, so if two matches occur during the same rating period, they would share the same t. A dataset is an ordered set of N match tuples $\mathcal{D} = \{(t_i, c_{i,a}, c_{i,b}, y_i)\}_{i=1}^{N}$.

We consider rating systems parameterized by $\boldsymbol{\theta}$. In the simplest case of the Elo rating system, $\boldsymbol{\theta} \in \mathbb{R}^C$ is a vector of ratings, one for each competitor. However other rating systems may use other parameters such as a representations of variance, volatility, running statistics, or higher dimensional vectors encoding game dynamics.

A rating system implements two functions, `predict` and `update`.

- `predict` takes as input the competitors $c_{i,a}$ and $c_{i,b}$, the time t_i, and the parameters from the previous rating period $\boldsymbol{\theta}_{t-1}$, and returns the predicted outcome $\hat{p}_i$ of match m_i. That is, the predicted probability that $c_{i,a}$ will win.

$$\hat{p}_i = \texttt{predict}(c_{i,a}, c_{i,b}, t_i, \boldsymbol{\theta}_{t-1})$$

- `update` takes as input the set of matches $\mathcal{M}_t = \{(t_i, c_{i,a}, c_{i,b}, y_i) \mid t_i = t\}$ occurring at time t, and the previous parameters $\boldsymbol{\theta}_{t-1}$, and produces the updated parameters $\boldsymbol{\theta}_t$.

$$\boldsymbol{\theta}_t = \texttt{update}(\mathcal{M}_t, \boldsymbol{\theta}_{t-1})$$

The parameter fitting and evaluation takes place simultaneously in a single pass over the dataset. We iterate one rating period at a time, first making predictions for each match in the period, then using the full match information including outcomes to update the parameters.

This setup is referred to as "online" or "time dynamic" and contrasts with methods which fit ratings to entire datasets at once as in Bradley and Terry (1952) or update ratings based on all previous data as in Coulom (2008).

3 Related Work

3.1 Rating Systems Evaluation

In early rating systems research, newly introduced methods were sometimes evaluated based on the agreement of their predictions with observed outcomes in chess (Elo, 1978), the ability of the systems

to recover underlying parameters of synthetically generated data (Glickman and Jones, 1999), and the degree to which ranked lists of competitors agreed with expert opinions (Glickman, 1999; 2001).

As the field has progressed, there has been a shift toward experiments which directly compare different rating systems and measure which make more accurate predictions using quantifiable metrics on the same datasets. Weng and Lin (2011) evaluated their methods on the same Halo dataset introduced by Herbrich et al. (2006) and compared prediction error rates. Szczecinski and Tihon (2023) compare their methods with Glicko and TrueSkill on synthetic data and with Elo on hockey data measuring the log loss. Kovalchik (2016) used tennis data and compared Elo to other types of match prediction methods such as point based models, regressions using player features, and bookmaker odds. The most extensive comparisons performed to date were done by Duffield et al. (2024), who perform comparisons of Elo, Glicko, TrueSkill, Simplified Kalman Filter, and others on chess, soccer and hockey datasets also comparing using log loss.

3.2 Rating Systems and Esports

While some novel paired comparison research has been evaluated on esports data, such as the evaluations done by Chen and Joachims (2016a;b); Bertrand et al. (2023) on StarCraft II data, and the study by Bisberg and Cardona-Rivera (2019) on Elo hyperpameter tuning using Call of Duty data, the majority of rating systems applications to esports are done using rating systems with the goal of making accurate predictions on esports matches, rather than using esports data to gain understanding about rating systems. There are numerous reports and projects in which rating systems like Elo, Glicko, and TrueSkill are used as baselines or as features in more sophisticated machine learning pipelines for esports prediction (Wahlroos, 2018; Pradhan and Abdourazakou, 2020; Dehpanah et al., 2021; Xenopoulos et al., 2022; Edmond, 2023).

3.3 Public Natural Language Processing Benchmarks

We also take inspiration from the public benchmarks and leaderboards used in the field of natural language processing. There are several benchmarks such superGLUE (Wang et al., 2019) and BigBench (bench authors, 2023) which have been widely adopted, allowing for straightforward comparisons across a large number of models. There is also a recent interest in using rating systems such as Elo as evaluation metrics in NLP using human paired comparison judgements (Bai et al., 2022; Boubdir et al., 2023; Chiang et al., 2024). We also learn from the weaknesses of public NLP benchmarks where occasionally models overfit to the benchmarks making them poor indicators of overall quality. Esports data for evaluation can avoid this issue as there is a steady stream of new and unseen match data from new events which can be used for evaluation before any additional training or hyperparameter tuning is done.

4 Dataset Details

We collect StarCraft II from aligulac[1], a fan run website which records results, statistics, and their own ratings of professional StarCraft II players. League of Legends data was collected from Leaguepedia[2], a wiki dedicated to the League of Legends professional scene. For all other games, we collect data from Liquipedia[3], a community of wikis covering many esports and administrated by Team Liquid, a professional esports organization. On the aligulac website, there is a form where contributors submit match results which are then made accessible by their API. The Leaguepedia and Liquipedia wikis have processes in place where editors update tournament pages with results using templates for matches and brackets. When the edits are made, the result data is automatically processed and imported to a database which is accessible by an API. The Liquipedia and Leaguepedia wikis and all data derived from them including EsportsBench are under a CC BY-SA 3.0 license.

[1] aligulac.com
[2] lol.fandom.com
[3] liquipedia.net

Table 1: Dataset Statistics

Game	First Year	# Matches Train	(Test)	# Competitors	Draw Rate
StarCraft I	1998	82,028	9,954	12,287	0.0069
StarCraft II	2010	411,030	22,343	22,517	0.0048
WarCraft III	2002	105,764	9,897	11,034	0.0060
Super Smash Bros. Melee	2004	372,332	19,482	40,437	0
Super Smash Bros. Ultimate	2018	242,245	25,030	40,151	0
Tekken	2013	37,217	13,772	10,114	0
Street Fighter	2010	56,294	14,956	12,831	0
King of Fighters	2012	13,003	3,568	3,653	0
Guilty Gear	2015	14,290	5,383	5,717	0
Tetris	2018	4,226	1,401	616	0.0002
FIFA	2002	17,285	9,168	4,011	0.0651
Rocket League	2015	125,543	22,353	28,609	0.0030
Dota 2	2011	66,330	6,691	7,690	0.0777
League of Legends	2011	104,737	17,806	12,829	0.0261
Counter-Strike	2001	170,003	19,302	25,597	0.0177
Call of Duty	2009	13,271	2,352	2,920	0.0033
Halo	2004	11,862	3,560	3,206	0.0056
Overwatch	2016	25,159	5,675	7,870	0.0080
VALORANT	2020	46,546	15,999	15,906	0.0106
Rainbow Six: Siege	2016	48,782	12,603	11,695	0.0260

4.1 Processing

The data covers a wide range of competitions, from world championships played in front of live crowds for millions of dollars, to weekly online tournaments and high school competitions. We opt to retain as much of the data as possible, filtering out data only to preserve correctness and competitive integrity. We filter out rows under the following conditions.

- The data has an empty or invalid entry in the date, competitor, or result fields.
- The two competitors have the same name/id. This can occur rarely in the cases of incorrectly entered data or non-unique names.
- The format is not standard. For example the occasional two-on-two matches in games which are usually one-on-one like StarCraft.
- The match is a show-match or another non-serious competition where the competitors' motivations are for entertainment rather than winning.

4.2 Statistics

In some cases, we elect to combine multiple games in a series, for example Halo, Halo 2, etc, into a single dataset. We do this in cases where there is a continuity in the community of professional players and teams who switch to playing new versions when they are released. For the StarCraft and Super Smash Bros. series, even when new games were released, substantial portions of the communities preferred to continue to play the older versions leading to multiple parallel esports scenes. In these cases we have individual datasets for different games in the series. Additional information about genre, competition format and series is recorded in table 3.

All datasets include data from the first year in which there are at least 100 rows, until the end of March 2024 with the final year of that range serving as the test set. Since the games cover different time ranges, and the relative popularity of games changes over time, there are different train/test ratios for each dataset. We choose this cutoff over splitting by percentage of rows or percentage of rating periods because this scheme naturally allows for the future additions of new matches on a quarterly basis with a 3 month shift to the test start range, while still allowing a standardized cutoff for comparisons on the static dataset snapshots.

Additional statistics about the distributions of matches per year, and distributions over player activity levels are shown in Figures 1 and 2. A common pattern is a relatively small set of strong, popular, and established players/teams have high activity levels and a very long tail of competitors who are seen only a few times or once. This is especially prominent in games with large open bracket tournaments where many entrants are eliminated in the first round and never appear again in the dataset. While the scales are different, Figure 2 shows that all of the datasets demonstrate this power law behavior.

4.3 Limitations

4.3.1 Data Correctness

As the underlying source for these datasets is data entered in wikis largely by volunteer fans, it is liable to standard data entry error patterns. The most common of these is typos in the entries of names, dates, and results. Errors in the names lead to multiple entities in the datasets both corresponding to the same real player or team. This leads to less accurate ratings and predictions as the full set of results is not encapsulated in the ratings for either of the entities. As player and team strength can vary over time, incorrect date information causes comparisons to be incorporated into the ratings at times where their ratings poorly reflect their strength, and in the worst case, it can transmit future result information back in time constituting a form of test set leakage. Errors in reported scores and results are less common but can directly cause the ratings to be less accurate representations of underlying strength.

Throughout the data curation process we detected many instances of these errors and contributed several thousand edits to the underlying wiki pages correcting them. Nevertheless, it is a near certainty that this category of error still persists in the published dataset and experimenters should be aware of these patterns. It is also our hope that having more eyes on these datasets can help uncover more errors allowing us and others to correct them in future releases.

4.3.2 Data Biases

There are biases present in these datasets due to the nature of their collection and entry. The primary bias is representation bias. Esports competitions are often segmented by region or level. In regional segmentation, regular competitions occur within a geographic region, with the top finishers from each region qualifying for less frequent international or worldwide events. There are also leagues or tournament systems that operate in a tiered structure with higher and lower levels of competition, including periodic opportunities for promotion and relegation across levels.

With data being collected and entered by volunteers, there is more attention and effort spent on competitions from the most popular regions and top levels. This results in the data from lower levels and less popular regions, often economically disadvantaged areas, being less complete and accurate. Missing and incorrect data can result in lower ratings for these competitors than their underlying skill would warrant.

Experimenters should also be mindful of data entry positional bias. In games like League of Legends and Dota 2, the competitor order can indicate faction or side, affecting gameplay. Other games feature symmetric gameplay where competitor order is arbitrary. Often, data entry volunteers may place the winner first by default. For instance, in Super Smash Bros. Melee, winners are listed first 58% of the time, despite order being irrelevant. While the rating difference methods used in this work are unaffected by order, other machine learning methods may incorrectly exploit this.

4.3.3 Data Granularity

There are two areas in which the coarse granularity of the data results in noise or loss of accuracy. The first is that while the data sources often contain date and time information, the time portion is not always reflective of reality. In particular it is often empty when editors only enter the date, or sometimes the back-end systems add time information based on the bracket structure to ensure that sorting results in a correct ordering, but with the times not being accurate, for example adding one additional minute to the time for each round in a bracket. For this reason in the final datasets we only include the date and not the time. Matches within the same date are not guaranteed to be ordered so all experiments using these datasets should use rating periods of no shorter than 1 day,

and make predictions independently for all matches within the same day to avoid possible leakage of information about future events.

Each row contains an outcome value of 1, 0.5 or 0 representing win, draw, or loss for the first competitor. However, rows may represent different formats. A common tournament format is for the regular season or early rounds to be best of 1, 2, or 3 matches, while for the playoffs or final rounds they switch to best of 5, or 7. Each row of the EsportsBench datasets represents a *match*, but may represent different numbers of *games* depending on the format and score.

5 Experiments

5.1 Methodology and Metrics

The goal of our experiments is to perform initial, representative benchmarks of current rating systems to gain insights into the best rating systems for esports data and the conditions in which they are better or worse as well as to provide baselines upon which future works can compare and build.

5.2 Models

We consider a variety of rating systems for our experiments, including classic popular methods as well as newer, highly general approaches. We implement the rating systems in python and publish the implementations separately as the open source package `riix`. (Thorrez, 2024).

Elo. The baseline is the Elo rating system. It estimates the outcome probability using a logistic function of the difference in competitor ratings, and updates the ratings based on the discrepancy between the predicted score and the outcome value. Let $\boldsymbol{\theta}$ be the vector of Elo ratings.

$$\hat{p}_i = \frac{1}{1+\exp^{-\alpha(\theta_a-\theta_b)}}, \text{where } \alpha = \frac{\log(10)}{400} \tag{1}$$

$$\begin{aligned} \theta'_a &= \theta_a + K(y_i - \hat{p}_i) \\ \theta'_b &= \theta_b - K(y_i - \hat{p}_i) \end{aligned} \tag{2}$$

K is a hyperparamter controlling the magnitude of the updates and can be viewed as the learning rate when the update is seen as a gradient step Balduzzi et al. (2018); Szczecinski and Tihon (2023); Morse (2019).

Glicko and Glicko2. The Glicko models (Glickman, 1999; 2001) are Bayesian extensions of Elo which maintain mean and variance parameters for each competitor. In addition to the variance, Glicko2 also tracks a volatility parameter for each competitor which controls how quickly their variance can change with time and new results. The Glicko models also perform updates based on aggregating all matches in the rating period. The rest of the rating systems are defined at a per-match level.

TrueSkill. TrueSkill was introduced in order to extend Elo beyond the one-on-one setting. It is also a Bayesian method and differs from Glicko in that it uses the Gaussian distribution rather than the logistic, and it fits parameters using factor graphs. In the one-on-one case the update equations simplify to a closed form.

Weng & Lin (W&L). The methods introduced in Weng and Lin (2011) Generalize and simplify TrueSkill. There are two variants, one that uses the logistic distribution, called Bradley-Terry (BT) and one that uses Gaussian distributions, called Thurstone-Mosteller (TM). The update equations are closed form even in the team setting. These methods rely on Stein's lemma in their approximation of the gradient and hessian of the likelihood.

Multidimensional Elo (mElo). Balduzzi et al. (2018) introduced a multidimensional extension of Elo which learns a Schur decomposition based on the game dynamics in order to model non-transitive cycles such as rock-paper-scissors interactions. In addition to the ratings, mElo maintains a k-dimensional vector for each competitor representing that competitor's strategy. Both the ratings and the strategy vectors are used when making predictions.

Generalized Elo (GenElo). Ingram (2021) introduces GenElo as a more simple Bayesian extension of Elo than Glicko, by assuming equal and constant variance for all competitors and only fits mean parameters. It can be derived by taking a single Newton step optimizing the log posterior.

Simplified Kalman Filter (vSKF). Szczecinski and Tihon (2023) provide a very general formulations of online rating systems building upon Kalman filter theory. These methods also come in Bradley-Terry and Thurstone-Mosteller variants. We consider the vector covariance version as it requires only one additional parameter per competitor in addition to the rating mean.

Variance Incorporated Elo (vElo). Hua et al. (2023) builds upon both Weng and Lin (2011) and Ingram (2021) generalizing Elo, maintaining both mean and variance parameters, and using the Laplace approximation for estimating the posterior. While the main focus is on incorporating tennis surface covariates, they provide an outcome only variant as part of the derivation which we use.

5.3 Hyperparameter Optimization

The methods we consider have different numbers of hyperparameters and varying sensitivity to changes in them. In order to perform a comparison which is fair to all methods, uses reasonably strong hyperparameters, and is computationally feasible, we use uniform random sampling for the hyperparameter values for all methods following Bergstra and Bengio (2012). We do this in two stages, first we use very broad ranges for the hyperparameter values roughly an order of magnitude lower and higher than values commonly used and suggested by the authors. Then we do a more fine grained sweep centered around the best values identified in the first stage. If the broad sweep identified x as the best value, the fine sweep samples in the range $[0.75x, 1.25x]$. We perform both sweeps with 1000 samples, and separately for each dataset. The full set of hyperparameters, ranges, and optimal values are listed in Table C. We run all experiments on a single machine with an AMD Ryzen 9 3900x processor where running 1000 hyperparameter combinations for each rating system and dataset takes about 5 hours.

We use 7 day rating periods for all experiments in this work. Our preliminary experiments using both longer and shorter periods both resulted in numerical instability for some methods so we leave that topic to future studies.

When performing the hyperparameter sweep we select the values which minimize the log loss.

$$\ell = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{p}_i) + (1 - y_i)\log(1 - \hat{p}_i)$$

Recognize that while draws are rare in these datasets, this metric does handle them naturally. When $y = 0.5$ it is equivalent to half the contribution of a win and and half of a loss. We choose the log loss as the objective over accuracy as it is a more smooth function with more room to differentiate between model scores from different models. Additionally many of the models are developed under statistical frameworks for optimizing this log likelihood.

We split all datasets into train and test sets based on date, with the test set being the most recent year of matches. Note that this does result in different train/test ratios for different datasets. We choose hyperparameters using the log loss on the entire train set. We use this methodology over a three way train, validation, and test split since in each run, matches are never trained on before they are predicted, so the same dataset can perform the validation role. The code used for our experiments is available at `https://github.com/cthorrez/esports-bench`.

5.4 Results

We present the test metrics using the best hyperparameters in table 2. Note that while all of the rating systems examined can train on drawn matches, many like Elo cannot predict a match to be a draw. For fairness we report accuracy computed on only decisive matches in the test set.

For the majority of games, Glicko and Glicko 2 are the best predictors both in accuracy and log loss with a very small edge for Glicko 2. The slight advantage of Glicko 2 makes sense as it is almost identical to Glicko with the only difference being allowing the increase in variance to vary across competitors. Despite Glicko's reputation and proven track record it is nevertheless surprising that it

Table 2: Test log-loss and accuracy (excluding draws) of each rating system on each dataset. The best rating system for each row is in bold.

Game	Metrics	Elo	Glicko	Glicko 2	TrueSkill	W&L BT	W&L TM	mElo	vSKF BT	vSKF TM	GenElo	vElo
StarCraft I	Accuracy	0.6001	0.6159	0.6164	0.6170	**0.6213**	0.6188	0.6044	0.5980	0.6178	0.6145	0.6136
	Log Loss	0.6516	0.6414	**0.6413**	0.6421	0.6428	0.6434	0.6492	0.6515	0.6435	0.6451	0.6439
StarCraft II	Accuracy	0.7977	0.8001	**0.8013**	0.7986	0.7904	0.7941	0.7983	0.7975	0.7979	0.7966	0.7990
	Log Loss	0.4230	0.4142	**0.4135**	0.4183	0.4335	0.4274	0.4225	0.4229	0.4188	0.4209	0.4208
WarCraft III	Accuracy	0.7690	**0.7789**	**0.7789**	0.7754	0.7633	0.7673	0.7689	0.7680	0.7730	0.7725	0.7698
	Log Loss	0.4735	0.4621	**0.4620**	0.4675	0.4797	0.4745	0.4730	0.4736	0.4715	0.4724	0.4700
Smash Melee	Accuracy	0.7250	**0.7353**	**0.7353**	0.7302	0.7223	0.7300	0.7252	0.7261	0.7287	0.7284	0.7310
	Log Loss	0.5353	0.5231	**0.5229**	0.5297	0.5316	0.5292	0.5356	0.5326	0.5284	0.5315	0.5262
Smash Ultimate	Accuracy	0.6772	**0.6883**	0.6881	0.6848	0.6842	0.6854	0.6773	0.6766	0.6811	0.6785	0.6842
	Log Loss	0.5915	**0.5828**	**0.5828**	0.5853	0.5847	0.5845	0.5915	0.5888	0.5840	0.5857	0.5836
Tekken	Accuracy	0.6426	**0.6562**	**0.6562**	0.6517	0.6487	0.6485	0.6412	0.6449	0.6507	0.6482	0.6490
	Log Loss	0.6129	**0.6060**	**0.6060**	0.6080	0.6085	0.6074	0.6130	0.6121	0.6095	0.6112	0.6093
Street Fighter	Accuracy	0.6260	**0.6303**	0.6302	0.6301	0.6240	0.6251	0.6265	0.6260	0.6291	0.6291	0.6247
	Log Loss	0.6433	0.6404	**0.6403**	0.6418	0.6389	0.6406	0.6433	0.6412	0.6394	0.6428	0.6415
King of Fighters	Accuracy	0.6393	**0.6501**	**0.6501**	0.6467	0.6355	0.6378	0.6389	0.6396	0.6436	0.6442	0.6378
	Log Loss	0.6295	**0.6236**	**0.6236**	0.6246	0.6262	0.6271	0.6302	0.6272	0.6247	0.6264	0.6269
Guilty Gear	Accuracy	0.6159	0.6166	0.6169	0.6164	0.6138	0.6162	0.6172	**0.6174**	0.6126	0.6131	0.6142
	Log Loss	0.6387	**0.6368**	**0.6368**	0.6381	0.6383	0.6377	0.6388	0.6377	0.6395	0.6400	0.6377
Tetris	Accuracy	0.6271	0.6278	0.6271	**0.6406**	0.6221	0.6242	0.6263	0.6292	0.6370	0.6242	**0.6406**
	Log Loss	0.6078	0.6073	0.6074	0.6088	0.6178	0.6143	0.6102	**0.6069**	0.6101	0.6153	0.6073
FIFA	Accuracy	0.5992	0.6041	**0.6053**	0.6024	0.6015	0.6017	0.6002	0.5996	0.6027	0.6019	0.5998
	Log Loss	0.6624	**0.6613**	**0.6613**	0.6619	0.6617	0.6615	0.6625	0.6621	0.6629	0.6620	0.6631
Rocket League	Accuracy	0.6314	0.6430	**0.6432**	0.6414	0.6359	0.6364	0.6312	0.6330	0.6400	0.6376	0.6354
	Log Loss	0.6334	**0.6261**	**0.6261**	0.6272	0.6305	0.6311	0.6334	0.6312	0.6267	0.6276	0.6313
DotA 2	Accuracy	0.6191	**0.6323**	**0.6323**	**0.6323**	0.6156	0.6212	0.6185	0.6229	0.6305	0.6316	0.6241
	Log Loss	0.6542	**0.6447**	**0.6447**	0.6465	0.6502	0.6506	0.6542	0.6512	0.6448	0.6453	0.6515
League of Legends	Accuracy	0.6340	0.6355	**0.6356**	0.6336	0.6251	0.6275	0.6336	0.6341	0.6308	0.6311	0.6329
	Log Loss	0.6371	**0.6292**	**0.6292**	0.6313	0.6401	0.6375	0.6372	0.6343	0.6299	0.6303	0.6346
Counter-Strike	Accuracy	0.6290	0.6417	**0.6418**	0.6386	0.6315	0.6338	0.6289	0.6314	0.6404	0.6378	0.6338
	Log Loss	0.6473	**0.6375**	**0.6375**	0.6391	0.6437	0.6432	0.6473	0.6447	0.6388	0.6394	0.6444
Call of Duty	Accuracy	0.6361	0.6433	**0.6442**	0.6433	0.6399	0.6408	0.6374	0.6387	0.6408	0.6408	0.6412
	Log Loss	0.6283	0.6197	**0.6196**	0.6215	0.6295	0.6242	0.6283	0.6268	0.6221	0.6228	0.6237
Halo	Accuracy	0.6788	**0.6941**	0.6935	0.6921	0.6825	0.6896	0.6825	0.6842	0.6899	0.6870	0.6893
	Log Loss	0.5882	**0.5722**	**0.5722**	0.5764	0.5820	0.5825	0.5879	0.5867	0.5776	0.5805	0.5817
Overwatch	Accuracy	0.6266	0.6387	**0.6390**	0.6353	0.6199	0.6249	0.6263	0.6275	0.6336	0.6308	0.6314
	Log Loss	0.6291	**0.6177**	**0.6177**	0.6207	0.6389	0.6341	0.6291	0.6234	0.6190	0.6197	0.6271
Valorant	Accuracy	0.6177	0.6255	**0.6256**	0.6229	0.6182	0.6174	0.6181	0.6200	0.6221	0.6197	0.6170
	Log Loss	0.6408	**0.6369**	**0.6369**	0.6385	0.6444	0.6415	0.6408	0.6385	0.6382	0.6391	0.6410
Rainbow Six	Accuracy	0.6257	0.6370	0.6361	**0.6371**	0.6299	0.6317	0.6251	0.6265	0.6344	0.6300	0.6283
	Log Loss	0.6372	**0.6271**	0.6272	0.6289	0.6330	0.6324	0.6372	0.6350	0.6299	0.6318	0.6333
Mean	Accuracy	0.6509	0.6597	**0.6599**	0.6585	0.6513	0.6536	0.6513	0.6521	0.6568	0.6549	0.6549
	Log Loss	0.6083	0.6005	**0.6004**	0.6028	0.6078	0.6062	0.6083	0.6064	0.6030	0.6045	0.6049

still outperforms many more recent rating systems. It is a good sign that all methods outperform the Elo baseline though by varying degrees. Overall the methods which measure uncertainty give larger gains than mElo which models intransitivity. With the scale and level of noise in these datasets, there are less clear examples of rock-paper-scissors type cycles.

In these experiments the Glicko methods are the only ones which naturally make a single update per competitor aggregating over all of the data in the rating period. For all other methods, despite matches in the same rating period being considered simultaneous, the update equations are defined at the per-match level so the matches within the rating period are treated as sequential for updating (predictions are still made for all matches in the period before any updates are performed). This difference may result in less stable updates for the non Glicko methods.

In terms of methodology, most other works make predictions at the match or even game level updating after each data point, thus those comparisons with Glicko deprive Glicko of the impact of aggregative updates. We note that when experimenting with different rating period sizes, shorter periods can result in better predictions and such a pattern is feasible in the online gaming setting, however for esports, especially in the case of large tournaments, it is often not practical to keep track of all results in real time and incorporate them into live predictions.

6 Future Work and Conclusion

There is an expansive range of extensions to and experiments using these datasets ripe for future work. An obvious direction for future work inspired by the strong performance of the Glicko models would be the extension of the update functions of other rating system to doing batched updates over all matches in a rating period at once. Additionally the impact of rating period length warrants further investigation. In this work we elected to perform a simple sampling based approach to hyperparameter tuning, future work should investigate more sophisticated methods such Bayesian optimization or the direct optimization demonstrated in Ingram (2021).

Additional fields can also be collected such as in-game statistics, competitor ages, and nationalities in order to compare with methods which incorporate features beyond the outcome such as Minka et al. (2018). Score data is available for the majority of the matches in the EsportsBench datasets allowing for further investigation into methods leveraging margin of victory (Kovalchik, 2020; Ingram, 2021). For team based competitions, data for the players on each team could be added in order to evaluate the gain methods designed for team competitions such as Herbrich et al. (2006); Weng and Lin (2011); Ebtekar and Liu (2021) bring over the team level modeling. There are esports events of all sorts practically every day, this means that there is a steady stream of new data which can be collected and released allowing for the potential development of EsportsBench into a dynamic or near-real-time leaderboard.

In this work we introduce EsportsBench, a diverse suite of 20 esports competition datasets for benchmarking rating systems. We outline a framework for rating systems experiments including setup, hyperparameter tuning, model parameter fitting, and evaluation. Our experimental findings from comparing 11 rating systems on the 20 datasets provide a foundational baseline upon which further experiments can extend and improve.

References

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

David Balduzzi, Karl Tuyls, Julien Perolat, and Thore Graepel. Re-evaluating evaluation. *Advances in Neural Information Processing Systems*, 31, 2018.

BIG bench authors. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=uyTL5Bvosj.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.

Quentin Bertrand, Wojciech Marian Czarnecki, and Gauthier Gidel. On the limitations of the elo, real-world games are transitive, not additive. In *International Conference on Artificial Intelligence and Statistics*, pages 2905–2921. PMLR, 2023.

Alexander J Bisberg and Rogelio E Cardona-Rivera. Scope: Selective cross-validation over parameters for elo. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 15, pages 116–122, 2019.

Meriem Boubdir, Edward Kim, Beyza Ermis, Sara Hooker, and Marzieh Fadaee. Elo uncovered: Robustness and best practices in language model evaluation. In Sebastian Gehrmann, Alex Wang, João Sedoc, Elizabeth Clark, Kaustubh Dhole, Khyathi Raghavi Chandu, Enrico Santus, and Hooman Sedghamiz, editors, *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 339–352, Singapore, December 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.gem-1.28.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Shuo Chen and Thorsten Joachims. Modeling intransitivity in matchup and comparison data. In *Proceedings of the ninth acm international conference on web search and data mining*, pages 227–236, 2016a.

Shuo Chen and Thorsten Joachims. Predicting matchups and preferences in context. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 775–784, 2016b.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. *arXiv preprint arXiv:2403.04132*, 2024.

Rémi Coulom. Whole-history rating: A bayesian rating system for players of time-varying strength. In *International conference on computers and games*, pages 113–124. Springer Berlin Heidelberg Berlin, Heidelberg, 2008.

Arman Dehpanah, Muheeb Faizan Ghori, Jonathan Gemmell, and Bamshad Mobasher. Evaluating team skill aggregation in online competitive games. In *2021 IEEE Conference on Games (CoG)*, pages 01–08. IEEE, 2021.

Samuel Duffield, Samuel Power, and Lorenzo Rimella. A state-space perspective on modelling and inference for online skill rating, 2024.

Aram Ebtekar and Paul Liu. Elo-mmr: A rating system for massive multiplayer competitions. In *Proceedings of the Web Conference 2021*, pages 1772–1784, 2021.

Christian Edmond. Determining win-loss probability and round differential in professional valorant. 2023. URL https://scholarworks.calstate.edu/concern/projects/pk02cj680. Published on 2023-11-20.

Arpad E. Elo. The uscf rating system: A scientific achievement. *Chess Life*, pages 160–161, June 1961.

Arpad E. Elo. The proposed uscf rating system: Its development, theory, and applications. *Chess Life*, 22(8):242–247, August 1967.

Arpad E. Elo. *The Rating of Chess Players Past and Present*. Arco, New York, 1978.

Ludwig Fahrmeir and Gerhard Tutz. Dynamic stochastic models for time-dependent ordered paired comparison systems. *Journal of the American Statistical Association*, 89(428):1438–1449, 1994.

Mark E. Glickman. A comprehensive guide to chess ratings. *American Chess Journal*, 3(1):59–102, 1995a.

Mark E Glickman. The glicko system. *Boston University*, 16(8):9, 1995b.

Mark E Glickman. Parameter estimation in large dynamic paired comparison experiments (glicko). *Journal of the Royal Statistical Society Series C: Applied Statistics*, 48(3):377–394, 1999.

Mark E Glickman. Dynamic paired comparison models with stochastic variances (glicko2). *Journal of Applied Statistics*, 28(6):673–689, 2001.

Mark E Glickman and Albyn C Jones. Rating the chess rating system. *CHANCE-BERLIN THEN NEW YORK-*, 12:21–28, 1999.

Mark E. Glickman and Hal S. Stern. A state-space model for national football league scores. *Journal of the American Statistical Association*, 93(441):25–35, March 1998. Applications and Case Studies.

Mark E Glickman, Jonathan Hennessy, and Alister Bent. A comparison of rating systems for competitive women's beach volleyball. *Statistica Applicata-Italian Journal of Applied Statistics*, (2):233–254, 2018.

Ralf Herbrich, Tom Minka, and Thore Graepel. Trueskill™: a bayesian skill rating system. *Advances in neural information processing systems*, 19, 2006.

Hsuan-Fu Hua, Ching-Ju Chang, Tse-Ching Lin, and Ruby Chiu-Hsing Weng. Rating players by laplace's approximation and dynamic modeling. *International Journal of Forecasting*, 2023.

Martin Ingram. How to extend elo: a bayesian perspective. *Journal of Quantitative Analysis in Sports*, 17(3):203–219, 2021. doi: doi:10.1515/jqas-2020-0066. URL https://doi.org/10.1515/jqas-2020-0066.

Stephanie Kovalchik. Extension of the elo rating system to margin of victory. *International Journal of Forecasting*, 36(4):1329–1341, 2020.

Stephanie Ann Kovalchik. Searching for the goat of tennis win prediction. *Journal of Quantitative Analysis in Sports*, 12(3):127–138, 2016.

Tom Minka, Ryan Cleven, and Yordan Zaykov. Trueskill 2: An improved bayesian skill rating system. *Technical Report*, 2018.

Steven Morse. Elo as a statistical learning model. https://stmorse.github.io/journal/Elo.html, 2019. Accessed: 2024-05-22.

Sean Pradhan and Yann Abdourazakou. "power ranking" professional circuit esports teams using multi-criteria decision-making (mcdm). *Journal of Sports Analytics*, 6(1):61–73, 2020.

Leszek Szczecinski and Raphaëlle Tihon. Simplified kalman filter for on-line rating: one-fits-all approach. *Journal of Quantitative Analysis in Sports*, 19(4):295–315, 2023.

Clayton Thorrez. Riix: Efficient rating system implementations, 2024. URL http://github.com/cthorrez/riix.

Minna Wahlroos. Esports match analytics, 2018.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.

Ruby C Weng and Chih-Jen Lin. A bayesian approximation method for online ranking. *Journal of Machine Learning Research*, 12(1), 2011.

Peter Xenopoulos, William Robert Freeman, and Claudio Silva. Analyzing the differences between professional and amateur esports through win probability. In *Proceedings of the ACM Web Conference 2022*, pages 3418–3427, 2022.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 4.3.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 4.3.2.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments (e.g. for benchmarks)...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See Section 5.3.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Sections 5.3 and C.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] The methods we used are deterministic and the data is order dependent preventing bootstrap sampling.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 5.3.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 4.
   (b) Did you mention the license of the assets? [Yes] See Section 4 and the Data and Code section of the supplemental.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] See Data and Code section in supplemental.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See personal data section in the supplemental.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] See personal data section in the supplemental.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Game Info

Table 3 provides additional information about the games, including their genre, competition format and whether the dataset is comprised of multiple games in a series.

Table 3: Information about the each game

Game	Genre	Competition Format	Series
StarCraft I	real time strategy	individual	
StarCraft II	real time strategy	individual	
WarCraft III	real time strategy	individual	
Super Smash Bros. Melee	fighting	individual	
Super Smash Bros. Ultimate	fighting	individual	
Tekken	fighting	individual	✓
Street Fighter	fighting	individual	✓
King of Fighters	fighting	individual	✓
Guilty Gear	fighting	individual	✓
Tetris	arcade	individual	
FIFA	sports	individual	✓
Rocket League	sports	team (3 players)	
DotA 2	multiplayer online battle arena	team (5 players)	
League of Legends	multiplayer online battle arena	team (5 players)	
Counter-Strike	shooting	team (5 players)	✓
Call of Duty	shooting	team (4 or 5 players)	✓
Halo	shooting	team (4 players)	✓
Overwatch	shooting	team (5 or 6 players)	✓
Valorant	shooting	team (5 players)	
Rainbow Six: Siege	shooting	team (5 players)	

B Dataset Distributions

To better understand the makeup of the datasets, Tables 1 and 2 visualize the number of matches per year, and per competitor in each of the datasets. The distributions per year highlight many interesting patterns. Several datasets begin very flat and then gave steep rises for example King of Fighters and Halo, while one might think this is strictly representative of popularity, it is much more an artifact of how online record keeping for esports results has drastically improved over time. It is important to keep in mind that the data availability is not uniform through time in these datasets which may have consequences on which rating systems are the best. Both this detail and the power law competitor match frequency are examples of how different esports data is from sports data in which leagues often operate in a set schedule where each team plays the same number of game in the same amount of time.

Another interesting artifact visible from the matches per year charts is the impact of the COVID 19 pandemic on different games. For games like Smash Brothers, where there are large grassroots in-person , 2020 and 2021 were notable drops compared to 2019. Conversely in Rocket League and WarCraft 3, there was a movement towards hosting more online tournaments, and the lower production costs allowed there to be more events and matches total.

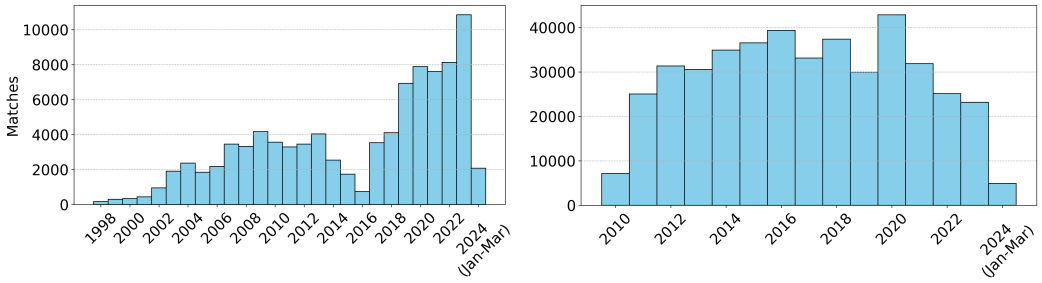
C Hyperparameters

Below is a full accounting of all hyperparameters for all rating systems and the ranges the broad hyperparameter sweep ran over. Note we do not consider the initial rating or mean hyperparameter since the considered methods are all sensitive only to the difference in ratings not their values. Note that some different hyperparameters share the same symbols when used in different rating systems, we are following the original authors' notations which sometimes overlap. As there are optimal values for each hyperparameter, dataset combination, we publish the optimal value configurations in the

Name	Rating Systems	Description	Values
k	Elo	magnitude of update	[3.2, 320]
RD	Glicko, Glicko2	initial rating deviation	[10, 1000]
c	Glicko	per time unit variance increase	[0.01, 100.0]
σ	Glicko2	initial per competitor volatility	[0.006, 0.6]
τ	Glicko2	volatility update constraint	[0.12, 1.2]
σ	TrueSkill, W&L	initial standard deviation	[0.8333, 83.33]
β	TrueSkill, W&L	rating difference implying 80% win prob	[0.4166, 41.66]
τ	TrueSkill, W&L	per game variance increase	[0.00833, 0.833]
v_0	vSKF	initial variance	[1e-6, 10]
β	vSKF	rating decay rate	[0.9, 1]
s	vSKF	scale, similar to the factor of $\log(10)/400$ in Elo	[0.1, 10]
ϵ	vSKF	per time unit variance increase	[1e-3, 0.1]
κ	W&L	minimum variance	[1e-5, 1e-3]
k	mElo	dimension of strategy vectors	1, 5, 10
η_r	mElo	magnitude of rating update	[1, 160]
η_c	mElo	magnitude of strategy vector update	[0.01, 1]
σ^2	vElo	initial variance	[1.0e-6, 10]
A	vElo	variance reduction factor	[1.0e-6, 1]
B	vElo	minimum variance	[1.0e-6, 10]
b	vElo	scale, similar to the factor of $\log(10)/400$ in Elo	[1.0e-6, 10]
σ	GenElo	shared constant standard deviation	[6, 600]

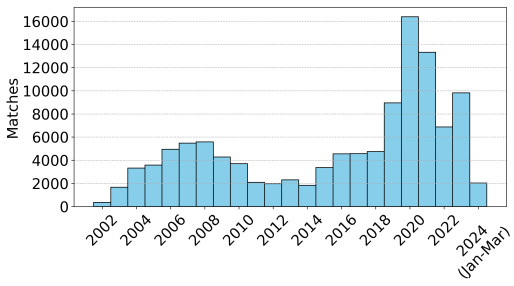
Table 4: Hyperparameter values, descriptions, and sweep ranges.

github repo at https://github.com/cthorrez/esports-bench/tree/main/esportsbench/experiments/sweep_results.

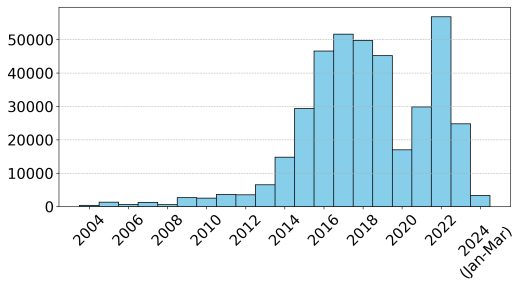


(a) StarCraft I

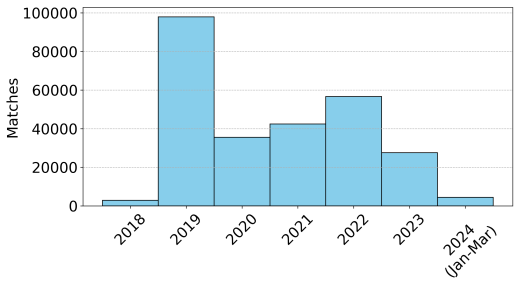
(b) StarCraft II



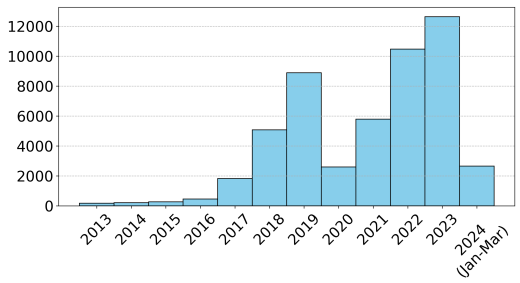
(c) WarCraft III



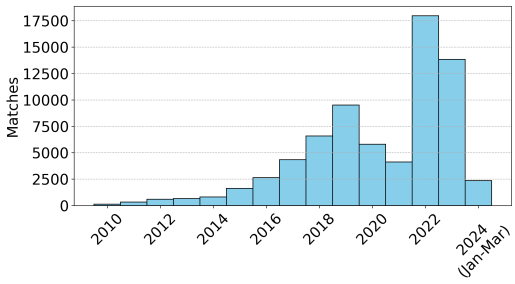
(d) Super Smash Bros. Melee



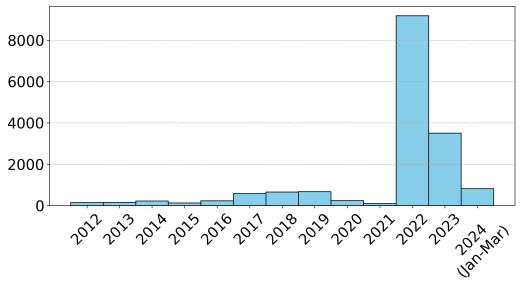
(e) Super Smash Bros. Ultimate



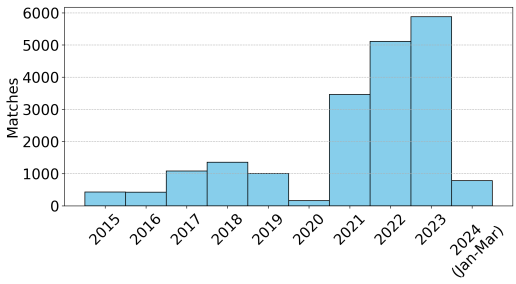
(f) Tekken



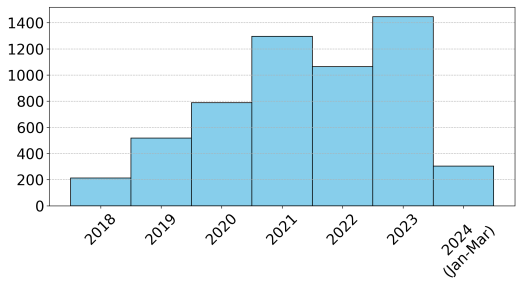
(g) Street Fighter



(h) King of Fighters

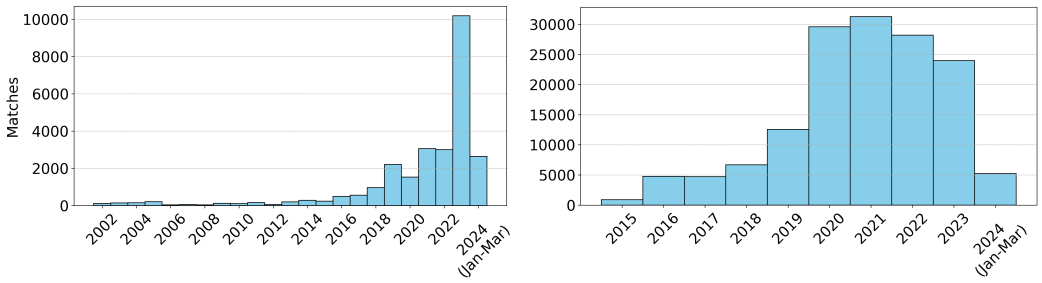


(i) Guilty Gear



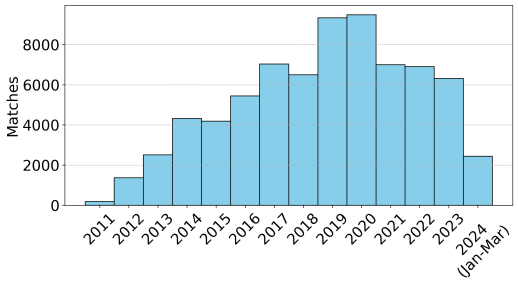
(j) Tetris

Figure 1: Number of matches per year for each game (1 of 2)

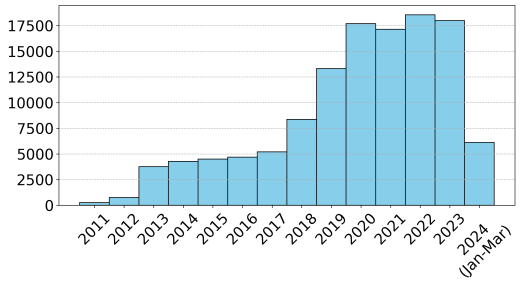


(k) FIFA

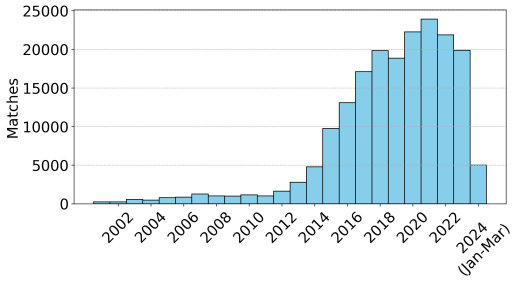
(l) Rocket League



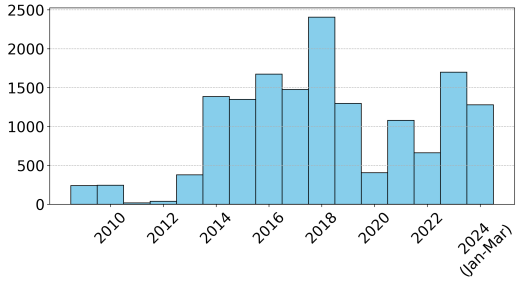
(m) Dota 2



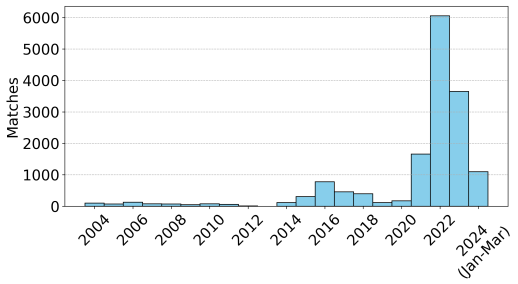
(n) League of Legends



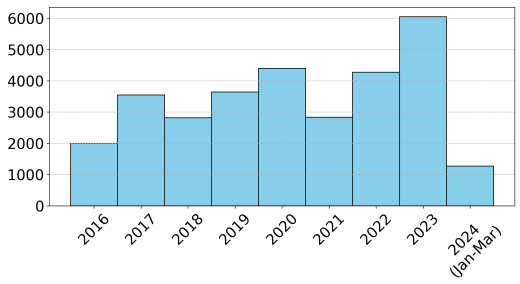
(o) Counter-Strike



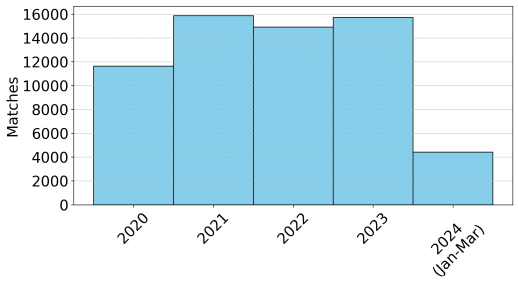
(p) Call of Duty



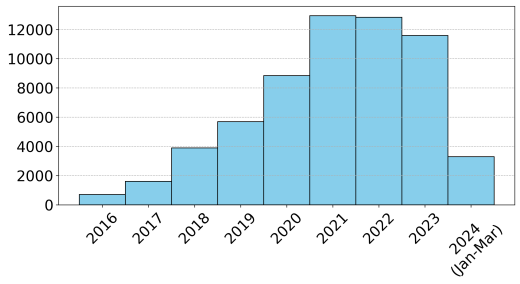
(q) Halo



(r) Overwatch

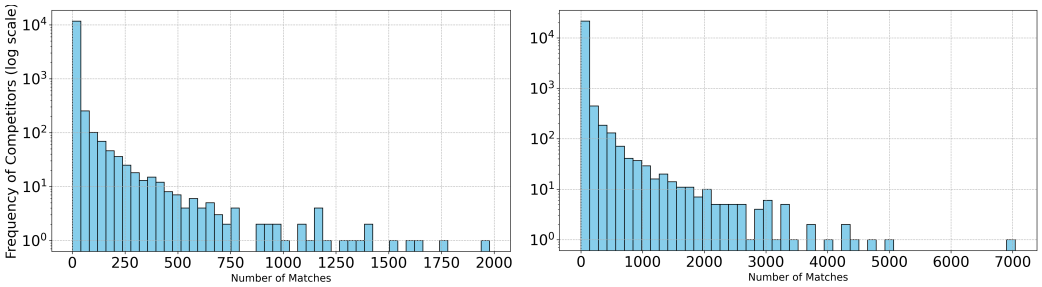


(s) VALORANT



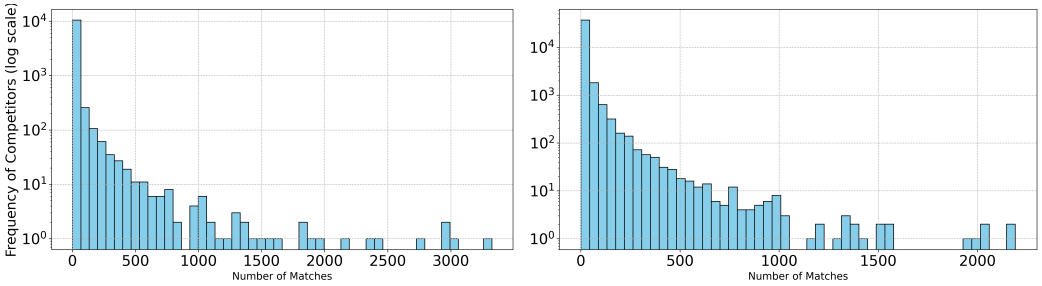
(t) Rainbow Six: Siege

Figure 1: Number of matches per year for each game (2 of 2)



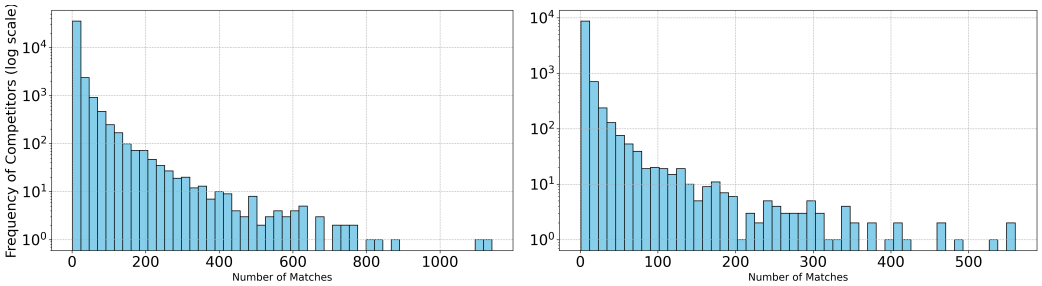
(a) StarCraft I

(b) StarCraft II



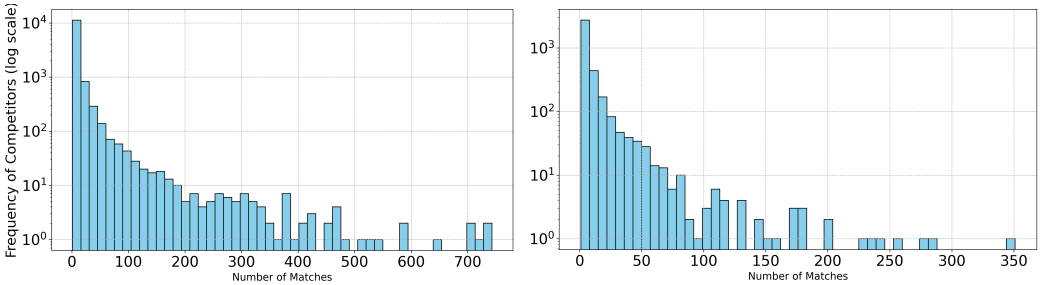
(c) WarCraft III

(d) Super Smash Bros. Melee



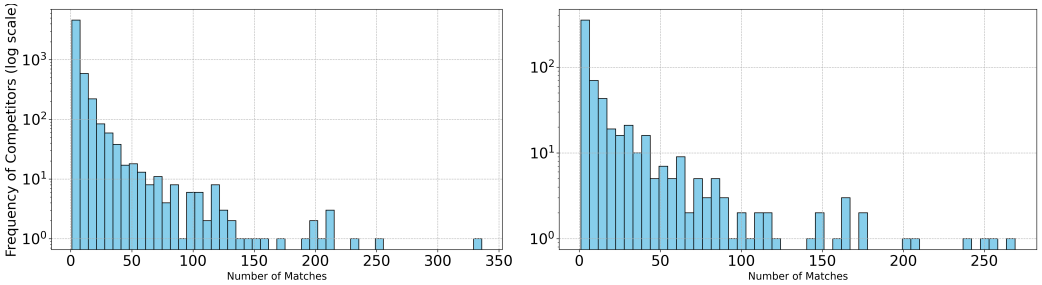
(e) Super Smash Bros. Ultimate

(f) Tekken



(g) Street Fighter

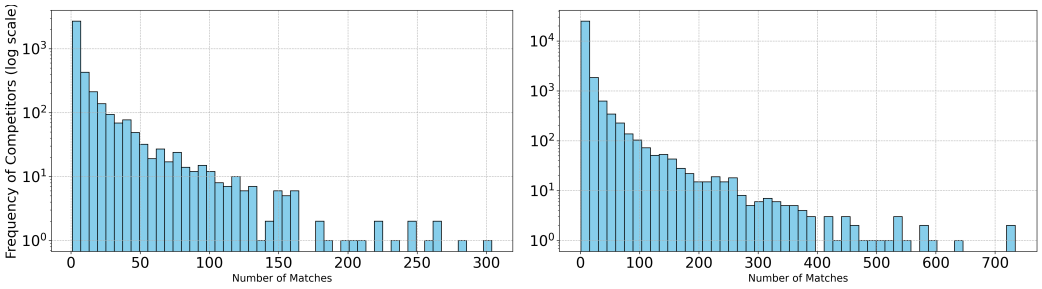
(h) King of Fighters



(i) Guilty Gear

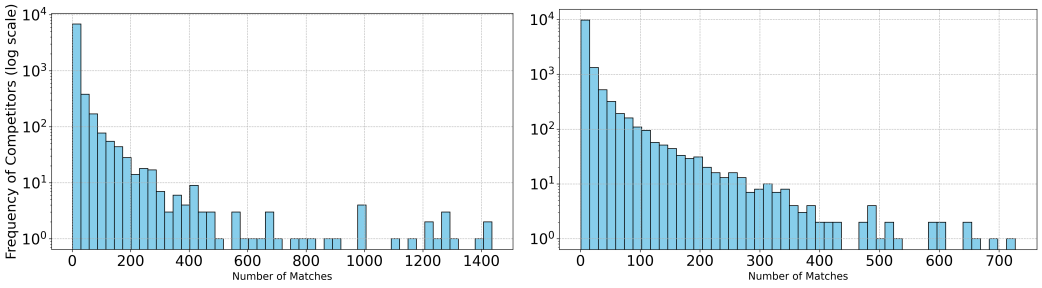
(j) Tetris

Figure 2: Frequency of matches per competitor in log scale (1 of 2)



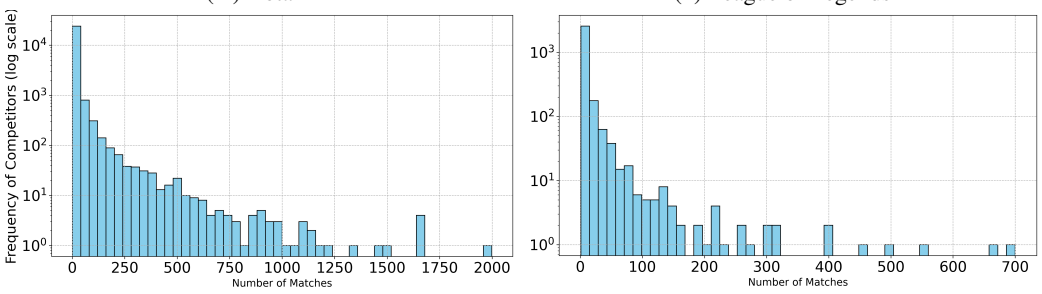
(k) FIFA

(l) Rocket League



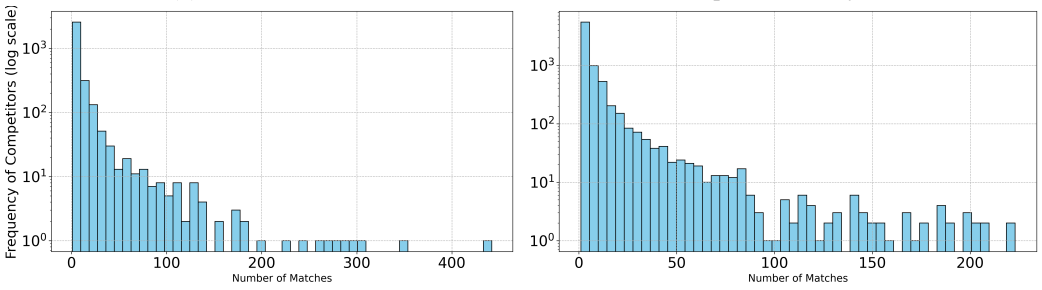
(m) Dota 2

(n) League of Legends



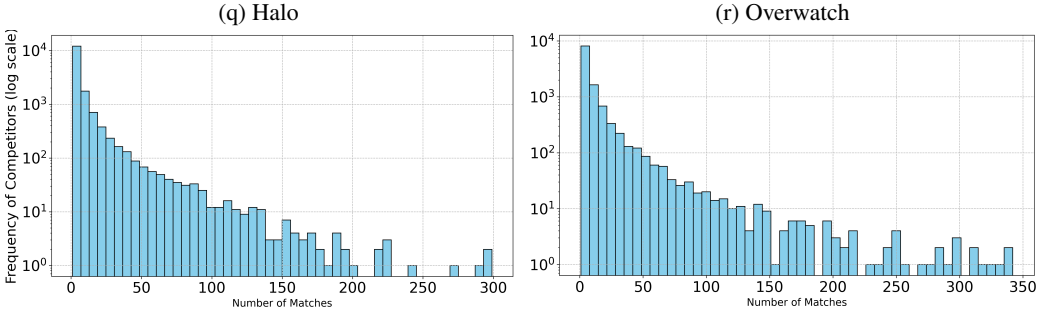
(o) Counter-Strike

(p) Call of Duty



(q) Halo

(r) Overwatch



(s) VALORANT

(t) Rainbow Six: Siege

Figure 2: Frequency of matches per competitor in log scale (2 of 2)