Team chemistry: the missing link in skill assessment for teams

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Colin Eugene DeLong

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

JAIDEEP SRIVASTAVA & LOREN TERVEEN

November, 2013

© Colin Eugene DeLong 2013
ALL RIGHTS RESERVED

Acknowledgements

I want to thank Katherine Pederson, who has been incredibly patient with me as I've burned the midnight oil writing this thesis, conducting experiments, creating figures and tables, and complained. Without her, I'm not sure how I would have been able to cross this finish line with my sanity intact.

I would also like to thank Nishith Pathak, Prasanna Desikan, Sandeep Mane, Kyong Shim, Gaurav Pandey, Ted Chow, and Jim Kang for their help, feedback, and support throughout graduate school.

Dedication

This thesis is dedicated to my brother, Trevor, and my friends Ryan Guanzon, Brent Carlson, and Ryan Avery, Angela Merritt, and Chris Kearns. Thank you for being there to listen and offer advice during the darkest days, and the brighter days, but especially for always believing in me throughout it all. I am forever grateful for your love and friendship.

Abstract

The task of assessing the skill of players and teams in games is an old problem spanning numerous disciplinary fields and can be traced back to foundational work from the early 20th century. However, in the past 15 years, the arrival and immense popularity of online multi-player gaming has kindled new interest in skill assessment due to the importance of ensuring that automatically-generated competitions (in a process called "matchmaking") between perhaps millions of candidate players and teams are fair - that each player or team competing against one another has a roughly equal probability of winning a given game. Poor matchmaking has the effect of discouraging less-skilled players from continuing to play, which, in games that are increasingly reliant on multi-player competition, is detrimental to a game's longevity and, therefore, its profitability. Beyond this problem, though, there exists a more general need to better account for attributes present in team-based games specifically, including the notion of "team chemistry" - a latent feature corresponding to the level of cohesion among teammates believed to impact the expected performance of teams not accounted for by the comparatively narrow lens of individual player skill alone.

In this thesis, we introduce a skill assessment framework which accounts for the effects of "team chemistry" using the performances of subgroups of players in teams. These subgroups therefore form the atomic unit to which skill ratings are assigned and maintained, standing in stark contrast to the existing practice of assigning skill ratings to individual players only. Further, existing skill assessment algorithms, such as Elo, Glicko, or TrueSkill, can be easily modified to be utilized as "base learners" for the maintenance of these subgroup ratings. The final estimated overall skill of a team is then computed as an aggregation of these subgroup skill ratings, and we describe a number of novel approaches for doing so. Through experimentation, it is shown that several of these aggregation approaches greatly improve the likelihood of correctly predicting the outcomes of unseen games, and we draw a number of interesting conclusions based on evaluations conducted on datasets from online multi-player video games and real-world sports.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Team-based games

Team-based games can be found in virtually every culture throughout the world, ranging from sports like soccer, football, and cricket, to online multi-player video games like Halo, Call of Duty, League of Legends, and Defense of the Ancients. Played and enjoyed by millions around the globe, team games are a reflection of the human desire to band together in collective competition, and as such, set themselves apart from contests between individuals with their requirement of teamwork as a fundamental component of a team's skill.

Teamwork is not the lone province of games, of course, and examples abound in many facets of modern society. Companies and other firms compete against each other for market share [2], [3], [4], hospitals often employ treatment plans based on group-care [5], [6] [7], and cities and governments coordinate their various constituent agencies as part of any effective disaster response strategy [8]. Common to all is the notion of teamwork, where "winning" requires high levels of coordination among groups of individuals.

In all of these contexts, the task of ascertaining a team's advantage over their opponents at any given point in time is a crucial task and often difficult to estimate. This continues to be a challenging open research problem in games, with well-known examples in sports like baseball where statistics abound for just about every in-game activity. It is now common to use Sabermetrics to build teams whose likelihood of winning a certain number of games in a season has a sound statistical basis [9], an approach which

has quickly spread to other sports [10], [11], [12].

The rise of online multi-player games has put the task of skill assessment front and center for game developers and publishers, wherein the long-term success or failure of a title is linked, in part, to the ability of players to find similarly-skilled teammates and opponents to play against. "Matchmaking", an automated process used to match players together for an online game, depends on accurate estimations of player skill at all times in order to reduce the likelihood of imbalanced matches. If one player or team is far superior to their opponents, the resulting game can frustrate less-skilled players and, over time, potentially lead to customer churn.

The question of who has the advantage is a challenge that persists. In the last several decades, however, vast increases in storage capacity and computational performance have opened the door to potentially more powerful skill assessment approaches. And with these advances, the granularity of in-game data tracked has reached a point where few actions are left un-recorded, resulting in a rich dataset for analysis.

1.2 Skill assessment

1.2.1 Overview

At a high level, skill assessment is the process of estimating the "worth" of players and/or teams, relative to the population of other players/teams, based on data culled from previously-played games. Similar to classification techniques, offline and online variants exist, though for skill assessment, research tends heavily toward those which are online due to the constraints placed on them in an operational environment. The work of this thesis is in line with that tendency and therefore concerned with online approaches, a setting in which the core assessment problem is more difficult to tackle because of the trade-offs made to ensure its real-world viability.

Skill assessment often comes down to a trade-off between the amount of game information used as features, and the computational feasibility in a real-world setting and generalizability across different games. In the world of electronic games especially, the dominant strategy is to use as little game-specific data as possible due to the sheer number of games occurring simultaneously, on the order of hundreds of thousands to

millions for the most popular games. Coupled with the need for reusability across different games, matchmaking algorithms are most often using only the data which is common to any form of competition so skill updates can be made after every game, forming the general case.

1.2.2 The general case

Consistent with other online learning frameworks, in the general case games are observed one at a time and skill ratings are updated after each game. Existing skill ratings are used to estimate the probability that one team defeats the other, and the team with probability of winning ≥ 0.5 is predicted the winner. The observed outcome of the game is compared to the prediction and the skill rating is updated to take into account the amount of confidence in the outcome asserted by the prediction, i.e., the distance from $p = 0.5$. Put differently, the size of the "correction" made to any skill rating is proportional to how correct or incorrect it was in predicting who would win the game.

As mentioned before, though, only the subset of the data available across *all* games is used to maintain this set of skill ratings. This subset consists of some or all of the following information:

- When the game was played
- Which players are competing
- Which teams each of the players are on
- Which team defeated the other
- The ratings of individual players at the time the game was played

This ensures the broad applicability of these algorithms to all games, regardless of mechanics, rules, or format. Additional data is inherently game- or genre-specific. For example, weather-related information assumes outdoor playing conditions which, although valuable if one team is well-versed competing in a typical physical environment, obviously do not apply to any games played indoors.

The score of the game is another feature which does not apply to all games, as the magnitude of victory is not used to ascertain which teams won or lost in games

organized around "placing" - the order in which the teams are ranked at the end of the game. This is most common in games with a single victory objective, like forcing the surrender of the other player in chess or StarCraft, or destroying the "Nexus" in League of Legends (and the "Ancient" in Defense of the Ancients 2).

Even the location of the game is not a feature relevant in all games. Until the advent of online multi-player video games, this could have been safely assumed to be essentially general case data, as the location of matches can often give one player or team the "home field" advantage, a widely-known factor in many stadium-oriented sports. Games occurring over the Internet, however, have no real "location" as such, and are therefore meaningless as variables in a skill assessment context. One forced exception is the usage of server location to act as a proxy variable for which player might experience the most network lag and, therefore, would have a proportionally limited ability to execute in-game commands in real-time. However, developers, in recognizing this issue, program matchmaking algorithms to vastly prefer matches located on or near each other with respect to the servers they're played on, greatly mitigating the prevalence of this potential imbalance. This general case is the hardest problem to solve in skill assessment, and remains unsolved after almost 100 years of continued research across a multitude of different fields.

1.2.3 The paired comparison estimation problem

In games, the question of how to rank (or provide ratings of) players is old, tracing its roots to the work of Louis Leon Thurstone from 1927 [13], wherein he proposed the "law of comparitive judgement", a means of measuring the mean distance between two physical stimuli, s_i and s_j. Mathematically, this represents the discriminal process, and in his measurement model, comparisons between pairs in a collection on the basis of some attribute. Thurstone, working with stimuli such as the sense-distance between levels of loudness, asserted that the distribution underlying each stimulus process is normally-distributed and that as such, the mean difference between the stimuli s_i and s_j can therefore be quantified in terms of their standard deviation:

$$s_i - s_j = x_{ij}\sqrt{\sigma_i^2 + \sigma_j^2 + 2r_{ij}\sigma_i\sigma_j} \tag{1.1}$$

In this notation, s_i is the psychological scale value of stimulus i, x_ij is the proportion of occasions on which the magnitude of stimulus i is judged to exceed the magnitude of stimulus j, σ_i is the "discriminal dispersion" of a stimulus i, and r_{ij} is the correlation between the discriminal dispersions of stimuli i and j. If the stimuli are assumed to be uncorrelated, then the previous equation can be rewritten as follows:

$$x_{ij} = \frac{s_i - s_j}{\sigma} \qquad\qquad \sigma = \sqrt{\sigma_i^2 + \sigma_j^2} \tag{1.2}$$

Thurstone's work laid the foundation for the formulation of Bradley-Terry-Luce (BTL) models in 1952 [14], a logistic variant of Thurstone's model which provided a rigorous mathematical examination of the paired comparison estimation problem [15], using taste preference measurements as its experimental example. In the BTL model, the logistic function was used to determine the "distance" between two objects:

$$x_{ij} \Rightarrow P(i > j) = \frac{e^{s_i - s_j}}{1 + e^{s_i - s_j}} \tag{1.3}$$

1.2.4 ELO

The BTL model framework provided the basis for the Elo rating system, introduced by Arpad Elo in 1959 [16]. Elo was himself a master chess player and developed the Elo rating system to replace the US Chess Federation's Harkness rating system [17] with one more grounded in statistical theory. Like Thurstone's model 1.2, the Elo rating system assumes each player's skill is normally distributed, where player i's expected performance is $p_i \sim N(\mu_i, \beta^2)$.

$$P(i > j) = \Phi(\frac{s_i - s_j}{\sqrt{2}\beta}) \tag{1.4}$$

Assuming p_i wins the game, the skill updates are as follows:

$$\Delta = \alpha\beta\sqrt{\pi}(1 - \Phi(\frac{s_i - s_j}{\sqrt{2}\beta})) \tag{1.5}$$

$$s_i^{'} = s_i + \Delta \tag{1.6}$$

$$s_j^{'} = s_j - \Delta \tag{1.7}$$

Notably, though, Elo also assumes players' skill distributions share a constant variance β^2, greatly simplifying the mathematical calculation at the expense of capturing the relative certainty of each player's skill.

1.2.5 The Glicko rating system

In 1993 [18], [19], Mark Glickman sought to improve upon the Elo rating system by addressing the ratings reliability issue in the Glicko rating system. By introducing a dynamic variance for each player, the confidence in a player's skill rating could be adjusted to produce more conservative skill estimates. However, the inclusion of this information at the player level also incurred significant computational cost in terms of updates, and so an approximate Bayesian updating scheme was devised which estimates the marginal posterior distribution $Pr(\theta|y)$, where θ and y correspond to the player strengths and the set of game outcomes observed thus far, respectively. Letting $s_i = N(\mu_i, \sigma_i^2)$:

$$\mu_i^{'} = \mu_i + \frac{q}{1/\sigma_i^2 + 1/\delta^2} \sum_{j=1}^{m} \sum_{k=1}^{n_j} g(\sigma_j^2)\{y_{jk} - E(y|\mu_i, \mu_j, \sigma_j^2)\}, \tag{1.8}$$

$$\sigma_i^{'2} = (\frac{1}{\sigma_i^2} + \frac{1}{\delta^2})^{-1} \tag{1.9}$$

Where

$$q = log(10)/400, \tag{1.10}$$

$$g(\sigma^2) = \frac{1}{\sqrt{1 + 3q^2\sigma^2/\pi^2}}, \tag{1.11}$$

$$E(y|\mu_i, \mu_j, \sigma_j^2) = \frac{1}{1 + 10^{-g(\sigma_j^2)(\mu_i - \mu_j)/400}}, \tag{1.12}$$

$$\delta^2 = \left[q^2 \sum_{j=1}^{m} n_j g(\sigma_j^2)^2 E(y|\mu_i, \mu_j, \sigma_j^2)\{1 - E(y|\mu_i, \mu_j, \sigma_j^2)\} \right]^{-1} \tag{1.13}$$

1.2.6 TrueSkill

With the advent of large-scale console-based multi-player gaming on the Microsoft Xbox in 2002 via Xbox Live, there was a growing need for a more generalized ratings system

not solely designed for individual players, but teams - and any number of them - as well. TrueSkill [1], published in 2006 by Ralf Herbrich and Thore Graepel of Microsoft Research, used a factor graph-based approach to accomplish this. Like Glicko, TrueSkill also maintains a notion of variance for each player, but unlike it, TrueSkill samples an expected performance p_i given a player's expected skill, which is then summed for all players on i's team to represent the collective skill of that team (an example is shown in Figure 1.1). This expected performance p_i is also assumed to be distributed normally, but similar to Elo, a constant variance is assumed across all players.

Of note, TrueSkill's summation of expected player performances in quantifying a team's expected performance assumes player performances are independent of one another. In the case of team games, especially those occurring at high levels of competition where team chemistry and cooperative strategies play much larger roles, this assumption may prove problematic in ascertaining which team has the true advantage a priori.

1.2.7 Other systems

Other variants of the aforementioned approaches have also been proposed. Coulom's Whole History Rating (WHR) method [20] is, like other rating systems such as Elo, based on the dynamic BTL model. Instead of incrementally updating the skill distributions of each player after a match, it approximates the maximum a posteri over all previous games and opponents, resulting in a more accurate skill estimation. This comes at the cost of some computational ease and efficiency, which the authors argue is still minimal if deployed on large-scale game servers. Others [21] have extended the BTL model to use group comparisons instead of paired comparisons, but also assume player performance independence by defining a team's skill as the sum of its players'.

In a study of adaptive pairwise tournaments, Beygelzimer et al. [22] set out to design a robust tournament algorithm. To choose the best player, the algorithm uses pairwise comparisons between players. The study states that TrueSkill models player performance as a normally distributed random variable that is centered around the skill level and that each pairing's outcome is determined by the outcomes of the corresponding random variables and that these assumptions do not fit their problem of modeling adversaries.

Birlutiu and Heskes [23] develop and evaluate variants of expectation propagation

techniques for analysis of paired comparison data by rating tennis players, stating that the methods are generalizable to more complex models such as TrueSkill. Menke, et al. [24], [25] develop a BTL-based model based on the logistic distribution, asserting that weaker teams are more likely to win than what a normally-distributed framework would predict. They also conclude that models based on normal distributions, such as TrueSkill, lead to an exponential increase in team ratings when one team has more players than another.

The field of game theory includes a number of related concepts, such as the Shapley value [26], which considers the problem of how to fairly allocate gains among a coalition of players in a game. In the traditional formulation of skill assessment approaches, however, gains or losses are implicitly assumed to be equal for all players given the limitation to win/loss/team formation history during model construction and evaluation. That is, no additional information is available to measure the contribution of each player to a team's win or loss.

1.2.8 The missing pieces

As mentioned and described previously, the general case is a very hard prediction problem. Even the best of these approaches may only achieve 70% accuracy on real-world datasets [1]. Further, team-based approaches must address "curse of dimensionality" issue somehow, sacrificing potentially useful information in order to produce an algorithm which is computationally feasible. That said, there are few approaches which can be said to both fit the general case framework and work for teams, and so there is territory yet to be explored.

The explicit assessment of skill for combinations of players, or subgroups, is a notable omission. As noted, for the general case occurring within a team-based game context, the data necessary to assign skill ratings to these subgroups, i.e., the composition of players on each of the competing teams for each game, is available. This data has not before been exploited in a skill assessment framework, instead limiting the maintenance of skill ratings to that of individual players, and estimate the collective skill of a team by summing together the ratings of all its players.

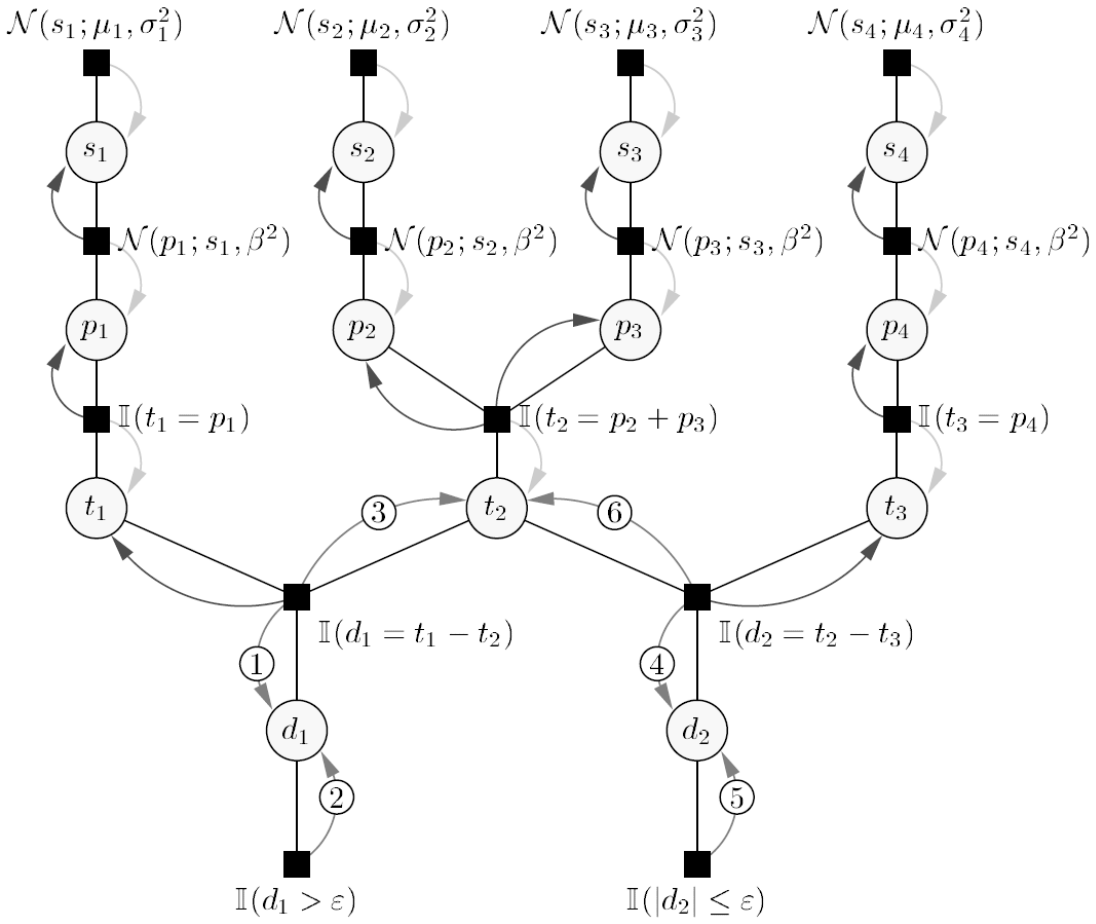


Figure 1.1: A sample factor graph for TrueSkill. In this graph, there are 3 teams. t_1 and t_3 have one player each, while t_2 has 2 players. Here, $t_1 > t_2 == t_3$. Note: this figure was taken from [1].

Chapter 2

A formalized model for team chemistry

2.1 Team chemistry

"Team chemistry" is a widely-held notion in team sports [27], [28] and is often cited as a key differentiating factor, particularly at the highest levels of competition. Though "team chemistry" resists a single definition (or name, as it is alternately referred to in related terms as cohesion or synergy depending on the context), one can think of it as the social dynamic arising in a group - or team - oriented towards accomplishing a common overall goal. And despite recent psychological research on group processes [29] stating that it is unclear as to whether team chemistry drives performance, or vice versa, it is nonetheless accepted as a key feature of successful teams in team-based sports and business. Or, to put it differently, the importance of team chemistry is not diminished by questions as to the process generating it in the first place.

It was Aristotle who once said (or was paraphrased as saying [30], [31]): "The whole is greater than the sum of its parts" [1] . Paraphrasing aside, this well-known tenet provides a useful framework for approaching the problem of modeling team chemistry.

[1] In the W. D. Ross translation, the quotation is "In the case of all things which have several parts and in which the totality is not, as it were, a mere heap, but the whole is something beside the parts, there is a cause", and in H. Tredennick's translation, it is "In all things which have a plurality of parts, and which are not a total aggregate but a whole of some sort distinct from the parts, there is some cause"

In fact, the quotation itself can be deconstructed in the context of teams and team chemistry:

- The "parts" are the individual players on a team
- The "whole" is the team
- The "more than the sum" is the notion of team chemistry, coordination, leadership, etc. that can positively - or negatively - impact the skill and subsequent observed performances of a team

The first two points are effectively self evident, but the third is less so. It is an assertion about the relationship between the "parts" and the "whole", and that the skill of a team is influenced by its collective chemistry. The question then becomes how, exactly, to model team chemistry's influence on a team's skill and, consequently, the effect it has when predicting which team is likely to defeat the other in competition.

Before describing the particulars, however, it is important to further frame this analysis by noting that "skill" and "performance" are different things. One can possess great skill and still occasionally perform poorly, and great performances are occasionally seen from less-skilled individuals or teams. In more formalized terms, "skill" is the latent variable approximated by skill assessment processes, while "performance" is something we observe that informs this approximation. As such, if a player or team's performances improve or decline over time, relative to that of their opposing teams, "skill" must itself be a moving target as well. It follows that good skill assessment approaches can explain most prior performances and accurately predict most future performances.

Given this, it is natural to consider team chemistry as another latent factor helping to explain performance. Following that line of thought, we might think of a team's skill as a function of the individual skills of each of the players on a team *and* their team chemistry. That is, if s_i is the skill of team i, and s_{ij} is the skill of the j-ith player on team i, and tc_i is the team chemistry of team i, then we might formulate a function such that

$$f(s_{ij}, tc_i) \rightarrow s_i \tag{2.1}$$

Unfortunately, this approach poses several large problems in an operational setting, the most pressing of which is its inability to address games where players change teams

frequently. Though not perceived as an issue of great import in real-world sports, this is the most common case in online multi-player video games, where players and teams are matched up via automated matchmaking algorithms. With team chemistry modeled as a separate latent factor to approximate, any new team composition faces the "cold start" issue wherein enough playing history must be observed in order to estimate the team's chemistry.

Another vexing problem is the approximation of team chemistry itself as a factor *separate* from skill. Not only is this misaligned with current psychological research [29], in the general case, it is difficult to imagine how team chemistry *could* be approximated without utilizing skill ratings as part of its foundation. That is, if we think of "positive" team chemistry as a team which performs beyond what their "sum of parts" skill ratings would suggest, then team chemistry cannot be separate from skill. Rather, a richer skill assessment framework accounting for team chemistry via its latent effects will simply find that it better explains the performances of a team relative to what would be expected from a "sum of parts" approach.

2.2 Generalizing the rankable entity from individuals to subgroups

In order to find a solution to this quandary, we first recall that existing skill assessment techniques maintain ratings for individual players, and that these ratings are used to estimate the collective skills of teams. It may be the case, however, that a subset of players on a team were teammates on a previous team and that whenever they are teammates, they play better. This is not uncommon in games of all sorts, and especially online multi-player games where many teams are composed of randomly-selected teammates and opponents - those who have "teamed" before and are capable of more efficient communication, coordination, unspoken role assignment, etc. are generally considered to have an advantage over a group of similarly-skilled individuals who have never played together. Two players who perform better when playing together is, in fact, the most basic unit in which team chemistry can be observed. It is therefore reasonable to ask *why* ratings aren't maintained for groups of players as well as individuals?

Returning to the second of the aforementioned "missing pieces" 1.2.8, if we theorize

that the skill of subgroups of teams is meaningful with respect to predicting the outcomes of games, then we can consider the skill ratings of subgroups of teams as clues regarding the overall cohesion of a team and, therefore, its overall skill. Put differently, we can model a team's skill as an aggregation of its component subgroup skill ratings, including subgroups which contain only individual players, and produce a model which incorporates team chemistry *not* separated from skill:

$$f(s_{i\bar{x}} \in 2^{S_i} \setminus \varnothing) \rightarrow s_i \tag{2.2}$$

In this formulation, $2^{S_i} \setminus \varnothing$ is the power set, minus the empty set, of a team's players and $s_{i\bar{x}}$ is the skill rating of a member subgroup. Critically, these ratings are largely algorithm-agnostic: with relatively minor modification, widely-used approaches like Elo, Glicko, and TrueSkill may be used to maintain the ratings for subgroups of any size, including those of individuals. Pseudocode for this operation is given in Figure 2.1.

Interestingly, the "sum of parts" approach of existing methods is a specific case of this more generalized framework wherein only the skills of subgroups having a single member are used in the estimation of a team's skill. But, since we now have much more, our focus turns to the problem of how to produce an estimate of a team's skill leveraging this "new" information.

2.3 Treating ratings for individuals and subgroups as "learners"

With the ratings of all subgroups of a team in hand, we can now formulate a strategy, or several strategies, for exploiting this information in an attempt to produce a better estimate of a team's skill. In this context, it is useful to think of a subgroup rating as partial evidence of the its skill, and chemistry if the subgroup contains more than one member. This can lead to several interesting use cases. For example, subgroups with overlapping membership may have different skill ratings, resulting in varying perceptions of the skill of those members in common. Similarly, we may observe that subgroups with overlapping membership may reinforce each other such that we can assume a greater degree of confidence in their respective ratings. As such, we can quite naturally develop a framework in which the team skill estimation problem can be cast in terms of ensemble

classification [32], considering each of the subgroup ratings as "learners" in the subset of game history shared by each of its members.

Though ratings exist for all subgroups of players on a team, it need not be the case that all the ratings be used to estimate a team's skill. A "sum of parts" strategy is one such approach and has worked well in practice for decades. It does so at the cost of subgroup-level skill and chemistry information, and so the process of deciding which "pieces" to use is inherently an assertion about what matters with respect to a team's skill.

We must also decide *how much* each of these pieces matter. A "sum of parts" strategy logically assigns equal weight to each individual player rating, but when working with subgroups of varying size and available game history, we may imagine cases in which this is neither appropriate or practical. For example, if we assume that the only game history which matters with respect to skill and team chemistry are the games shared by all members of the team, i.e. the games played by that particular team configuration, and therefore that the only skill rating which matters is the one corresponding to the subgroup of the team itself, then we would ignore entirely the ratings of all other subgroups. On the other hand, a "little of everything" approach can run into the practical difficulty of ranking the relative importance of aggregated subgroup skill ratings in terms of their hypothesized contribution to the "true" overall skill of a team. The question of "how much each piece matters" forms the basis for a number of possible aggregation techniques, which we will explore further in subsequent sections.

Figure 2.1: Pseudocode for alterations necessary for baseline skill assessment techniques to accommodate the ranking of subgroups.

input : A matrix of game data for games G played by players P; learner default rating p_0; maximum team size K
output: A matrix S of skill ratings for observed subgroups U

initialize $U \leftarrow P, S(U) \leftarrow p_0$;
for $g \in G$ **do**
  $t_w \leftarrow U(WinningTeam(g))$;
  $t_l \leftarrow U(LosingTeam(g))$;
  for $k = 1$ **to** K **do**
    /* Get k-sized subgroups in power set of team's players */
    $t_w^k \leftarrow \bar{x} \mid \bar{x} \in 2^{t_w} \cap |\bar{x}| = k$;
    $t_l^k \leftarrow \bar{x} \mid \bar{x} \in 2^{t_l} \cap |\bar{x}| = k$;
    /* Find the subgroups we're seeing for the first time... */
    $t_w'^k \leftarrow t_w^k \notin U$;
    $t_l'^k \leftarrow t_l^k \notin U$;
    /* ...and assign them default ratings */
    $S(t_w'^k) \leftarrow p_0$;
    $S(t_l'^k) \leftarrow p_0$;
    /* Compute the skill update for these k-sized subgroups according to learner */
    $\Delta_k \leftarrow f(S(t_w^k), S(t_l^k), learner)$;
    /* ...and update subgroups. Note: code generalized - updates may not be symmetrical for certain learners */
    $S(t_w^k) = S(t_w^k) + \Delta_k$;
    $S(t_l^k) = S(t_l^k) - \Delta_k$;
  end
end

Chapter 3

TeamSkill: ranking using team chemistry

The first logical step in understanding the effect of utilizing subgroup ratings to estimate the collective skill of a team is to learn the role that different-sized subgroups play in helping predict the outcome of a game. In doing so, we may begin to answer a number of questions about the viability of using this information and compare their performance to the baseline case: the "sum of parts" approach for individual players, or subgroups having one member.

This approach, though understandable from the perspective of minimizing computational costs and/or model complexity, is not well-aligned with either intuition or research in sports psychology [27], [28], [33]. Only in cases where the configuration of players remains constant throughout a team's game history can the summation of individual skill ratings be expected to closely approximate a team's true skill. Where that assumption cannot be made, it is difficult to know how much of a player's skill rating can be attributed to the individual and how much is an artifact of the players he/she has teamed with in the past.

In this naïve case, however, we will employ a "sum of parts" foundation, albeit with different-sized subgroups. The overall goal hasn't changed - use the ratings from each team's subgroups to estimate their respective skills. To do so, the summation of the ratings must be weighted to account for the size of the subgroups such that their

aggregation is scaled to the size of their teams. From 2.2, any subgroup having more than one member will have team chemistry as an additional latent effect, we refer to our family of approaches which utilize this information as "TeamSkill". TeamSkill-K is the first such approach, implementing the naïve case as just described.

3.1 TeamSkill-K

The overall approach is simple: for a team having K players, choose a subgroup size $k \leq K$, calculate the average skill rating for all k-sized player groups for that team using some "base learner" (such as Elo, Glicko, or TrueSkill), and finally scale this average skill rating up by $\frac{K}{k}$ to arrive at the team's skill rating. Recall that for $k = 1$, this approach is equivalent to simply summing the individual player skill ratings together. As such, TeamSkill-K can be thought of as a generalized approach for combining skill ratings for any K-sized team given player subgroup histories of size k.

Formally, let s_i^* be the estimated skill of team i and $f_i(k)$ be a function returning the set of skill ratings for player subgroups of size k in team i. Let each member of the set of skill ratings returned by $f_i(k)$ be denoted as s_{ikl}, corresponding to the l-ith configuration of size k for team i. Here, s_{ikl} is assumed to be a random variable drawn from some underlying distribution, typically a Gaussian ($s_{ikl} = N(\mu_i, \sigma_i^2)$). Then, given some k, the collective strength of a team of size K can be estimated as follows:

$$\begin{aligned} s_i^* &= \frac{K}{k} E[f_i(k)] \\ &= \frac{K}{k} \frac{1}{\frac{K!}{k!(K-k)!}} \sum_{l=1}^{\frac{K!}{k!(K-k)!}} s_{ikl} \\ &= \frac{K}{k} \frac{k!(K-k)!}{K!} \sum_{l=1}^{\frac{K!}{k!(K-k)!}} s_{ikl} \\ &= \frac{(k-1)!(K-k)!}{(K-1)!} \sum_{l=1}^{\frac{K!}{k!(K-k)!}} s_{ikl} \end{aligned} \tag{3.1}$$

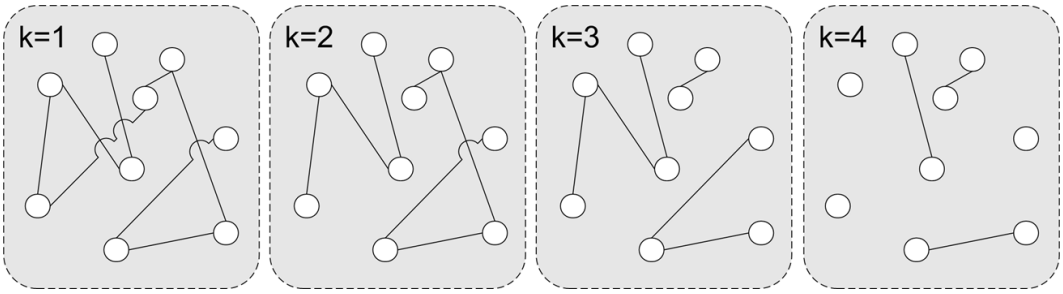


Figure 3.1: Graph-based illustration of game history availability vs. player subgroup specificity trade-off. Each node is a player subgroup of size k and each edge indicates the existence of shared history between two subgroups (i.e., they were opponents and/or teammates in the past).

3.1.1 Potential problems

In real-world scenarios, there are cases such that no game history exists for one or more subgroups of size k, and so, in the case of a Gaussian distribution, $N(\mu_{ikl}, \sigma^2_{ikl})$ may assume the default skill rating values for its corresponding base learner. This formulation would therefore make a possibly inaccurate assumption regarding the skills of new subgroups, but no more than existing approaches do for individual players who are playing (or more aptly in this context, for whom we are *observing*) their first game. This is a more localized version of the "cold start" problem, but for specific subgroups.

Though simple to implement and useful as a generalized approach for estimating a team's skill given ratings for player subgroups of size k, it is not without its issues. The choice of k introduces a potentially problematic trade-off between two desirable skill estimation properties - game history availability and player subgroup specificity. As the chosen k becomes larger, the subgroups can be expected to capture more of that particular team configuration's group cohesion effects, but at the cost of less available game history, and so a newly-formed team made up of individual players who have played many games on different teams may have subgroup ratings that do not reflect the "true" skill of that team until several games have been played. Alternatively, as k becomes smaller, subgroups capture lower-level interaction information between smaller subgroups of players, which, although having a more extensive set of game history to draw upon, limits or eliminates any collective effect attributable to team chemistry.

This trade-off can also pose issues in terms of the amount of "isolation" a set of

subgroup ratings has relative to other subgroup ratings. Consider a graph (see figure 3.1) where each player subgroup of size k is a node and there exists an edge when two such subgroups have played against and/or with one another in the past. When k is large, the graph is more likely to be composed of multiple connected components separated from one another rather than one large connected component, resulting in skill rating updates which are more localized than they would be for a smaller k value. The issue arises when two teams from different components play one another - their skill estimates prior to the first game aren't likely to be accurate since they share no history connecting one to the other (even through other teams), and so their skill ratings are essentially the result of games played within their "island" of competition. However, when k is small relative to K, less can be said about the strength of the team overall since subgroup interaction effects less likely to be captured the lower k is.

In real-world sports, we may find a version of this problem when two teams from different leagues play each other for the first time. For example, until 1997, American League and National League baseball teams only played each other during spring training and the World Series, and so while it was relatively easy to see which teams were strongest in each league, it was more difficult to get a sense of which league was inherently stronger *until* the World Series. This can also happen in online multi-player gaming, where player populations may be separated by geographic location (e.g., North America, Europe, China, and Korea in StarCraft 2) due to latency concerns. As a result, player skill ratings are largely the result of games played between other players from the same server, and so players from different regions having the same skill ratings may have vastly different "true" skill levels when analyzed from a global perspective. In TrueSkill-K, higher k values exacerbate this issue, potentially creating more "islands" due to fewer instances of inter-group game history.

3.2 TeamSkill-AllK

To address this issue in the hopes of finding a "best of both worlds" solution, a second approach was developed, called TeamSkill-AllK. Here, *all* available player subgroup information, $1 \leq k \leq K$, is used to estimate the skill rating of a team. Constructing the overall skill rating for a team in this way exploits the "cold start" advantages inherent to

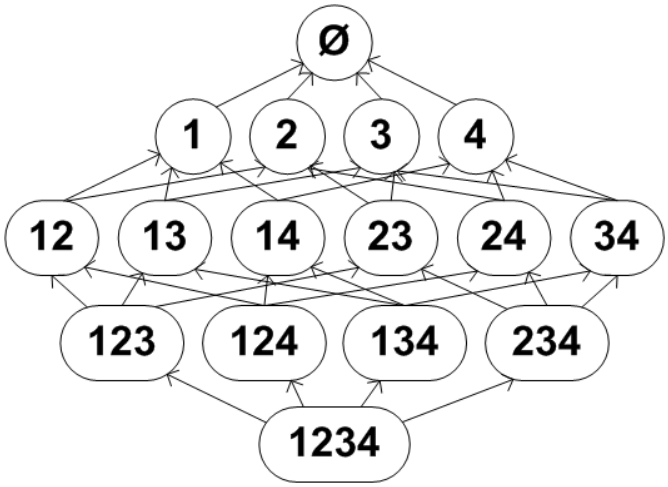


Figure 3.2: Recursive formulation of TeamSkill-AllK approach for a team of 4 players.

smaller k subgroups, mitigating the "island" effect for higher k subgroups until enough game history has been observed. As certainty grows regarding the skill rating of higher k subgroups, the overall team rating will be nudged in the direction of the these higher k subgroup ratings, either reinforcing the lower k subgroup ratings or diluting their effects via aggregation.

In TeamSkill-AllK, the general idea is to model a team's skill rating as a recursive summation over all player subgroup ratings, building in the $(k-1)$-level interactions present in a player subgroup of size k in order to arrive at the final rating estimate. Visualized, this approach forms a lattice structure, as can be seen in figure 3.2. In this approach, for instance, the skill attributed to node n_{123}, call it s^*_{123}, is a function of both the subgroup skill rating for n_{123} itself, its child subgroup ratings n_{12}, n_{13}, and n_{23}, and their child "subgroup" ratings n_1, n_2, and n_3 i.e., the players 1, 2, and 3. As in TeamSkill-K, default ratings are assigned to subgroups with no prior game history in order to ensure the recursion is viable.

This approach can be expressed as follows. Let s^*_{ikl} be the estimated skill rating of the l-ith configuration of size k for team i and $g_i(k)$ be a function returning the set of *estimated* skill ratings s^*_{ikl}, where $1 \leq l \leq \frac{K!}{k!(K-k)!}$ for player sets of size k in team i. When $k = 1$, let $s^*_{i1l} = s_{ikl}$. As before, let s_{ikl} be the skill rating of the l-ith configuration of size k for team i. In TeamSkill-AllK, if s_{ikl} has no prior game history and, as such, no

associated skill rating, then during aggregation, the base learner-specific default values, scaled to subgroup size k, are assumed. Finally, let α_k be a user-specified parameter in the range $[0, 1]$ signifying the weight of the k-ith level of estimated skill ratings. Then, for $k > 1$,

$$\begin{aligned} s^*_{ikl} = & \quad \alpha_k s_{ikl} + (1 - \alpha_k) \frac{k}{k-1} E[g_i(k-1)] \\ = & \quad \alpha_k s_{ikl} + (1 - \alpha_k) \frac{k}{k-1} \frac{\sum_{s^*_{ik-1l} \in g_i(k-1)} s^*_{ik-1l}}{|g_i(k-1)|} \end{aligned} \tag{3.2}$$

To compute s^*_i, let $s^*_i = s^*_{ikl}$ where $k = K$ and $l = 1$ (since there is only one player subgroup rating when $k = K$). This ensures that *all* player subset history is used in the recursion.

3.3 TeamSkill-AllK-EV

As mentioned, if no history is available for a particular subgroup in TeamSkill-AllK, default values are used instead in order to continue the recursion. In practice, however, cases where limited player subset history is available will produce team skill ratings largely dominated by default rating values, potentially resulting in inaccurate skill estimates. This is especially the case when first applying TeamSkill-AllK to a new dataset as little history is available for all player subgroups, regardless of subgroup size, which has the potential to overly diluting the effect of individual player ratings.

As such, another approach was developed, called TeamSkill-AllK-EV. The core idea of using all available player subgroup ratings was retained, but the new implementation eschews default values for all player subgroups save those of individual players (consistent with existing skill assessment approaches), instead focusing on the evidence drawn solely from game history. Over time, the performance of TeamSkill-AllK-EV and TeamSkill-AllK should be consistent with one another, but TeamSkill-AllK-EV should perform better in low-history situations, and at least as well as individual player ratings

alone. Re-using notation, TeamSkill-AllK-EV is as follows:

$$s_i^* = \frac{1}{\sum_{k=1}^{K} [h_i(k) \neq \varnothing]} \sum_{k=1}^{K} \frac{K}{k} E[h_i(k)]$$

$$= \frac{K}{\sum_{k=1}^{K} [h_i(k) \neq \varnothing]} \sum_{k=1}^{K} \frac{E[h_i(k)]}{k} \tag{3.3}$$

Here, $h_i(k) = f_i(k)$ where there exists at least one player subgroup rating of size k, else $\varnothing$ is returned and $E[\varnothing] \to 0$. $[h_i(k) \neq \varnothing]$ is an Iverson bracket, returning 1 if $h_i(k) \neq \varnothing$ and 0 otherwise, ensuring that the aggregate skill ratings for subgroups of size k, $E[h_i(k)]$, are weighted evenly when estimating the overall skill of a team. Assuming equal contributions from each set of subgroups of size k is obviously a naïve weighting strategy, but appropriate as there is no clear sense of how to better weight each set of k-sized subgroups at any given point in time.

3.4 TeamSkill-AllK-LS

In this context, it is natural to hypothesize that the most accurate team skill ratings could be computed using the largest possible player subgroups covering all members of a team. That is, given some player subgroup X and its associated rating, ratings for subgroups of X should be disregarded since they represent lower-level interaction information X would have already captured in its rating. An example of this approach is given in figure 3.3.

One obvious advantage to this approach is its speed, since this method prunes away from consideration ratings of subgroups derived from their previously-used superset subgroups. This approach is reminiscent of the Apriori algorithm [34], but in reverse:

- We begin with the largest possible subset of a team of size K, namely the subgroup of size $k = K$, and see if a rating already exists for that subgroup. If it does, then we are done as that is the largest subgroup covering all the team's players.
- If there is no such subgroup, then we set $k = k - 1$ and check for any subgroups that aren't themselves subsets of subgroups for which we already have ratings.
- This process is repeated until we have a set of subgroup ratings which covers all players on the team.

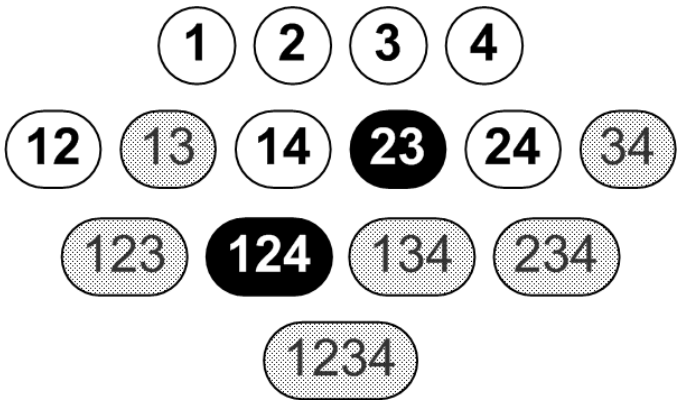


Figure 3.3: Example of TeamSkill-AllK-LS approach. Unshaded cells indicate that history is available for a given subgroup while lightly-shaded cells indicate that history is not available. The darkly-shaded subgroups are the largest subgroups covering all members of a team.

Formally, such an approach can be represented as follows:

$$s_i^* = \frac{K}{\sum_{m=K}^{1} {}^{m|\{h_i(m) \not\subseteq h_i(m<j\leq K)\} \neq \varnothing}} \sum_{k=K}^{1} {}^{E[h_i(k) \not\subseteq h_i(k<j\leq K)]} \tag{3.4}$$

Notably, this approach may have problems with newly-formed teams - after a single game, there will exist a rating for the subgroup of $k = K$, which by itself may be a bad approximation of a team's skill until a number of games have been observed. This mirrors the issues faced by TeamSkill-K where $k = K$.

3.5 Experiments

Evaluation is carried out on a carefully-compiled dataset consisting of tournament and scrimmage games between professional and semiprofessional Halo 3 teams over the course of two years (see Appendix A for more details). Halo 3 is a first-person shooter (FPS) game which was played competitively in Major League Gaming (MLG), the largest professional video game league in the world, from 2008 through 2010. With MLG tournaments regularly featuring 250+ Halo teams vying for top placings, heavy emphasis is placed on teamwork, making this dataset ideal for the evaluation of interaction effects among teammates.

The four proposed TeamSkill approaches were evaluated by predicting the outcomes of games occurring prior to 10 tournaments, comparing their accuracy to unaltered versions ($k = 1$, henceforth referred to as the baseline version) of their base learner skill assessment algorithms - Elo, Glicko, and TrueSkill. For TeamSkill-K, all possible choices of k for teams of 4 - the maximum number of players in a professional Halo 3 team - $1 \leq k \leq 4$, were used.

For each tournament, we evaluated each TeamSkill method using:

- 3 types of training data sets - games consisting only of tournament data, games from online scrimmages only, and games of both types.
- 3 periods of game history - all data except for the data between the test tournament and the one preceding it ("long"), all data between the test tournament and the one preceding it ("recent"), and all data before the test tournament ("complete").
- 2 types of games - all games i.e., the full dataset, and the subset of those which were considered "close" (i.e., prior probability of one team winning close to 50%).

In the case where only tournament data is used as training set data, the most recent tournament preceding the test tournament replaced the inter-tournament scrimmage data for the "long" and "recent" game history configurations. Similarly, "recent" game history when considering both tournament and scrimmage data included the most recent tournament. "Close" games were determined using a slightly modified version of the "challenge" method described in [1] in which the top 20% closest games were selected for one rating system and presented to the other (and vice versa). In this evaluation, the closest games from the baseline versions of each rating system (i.e., $k = 1$) were presented to each of the TeamSkill approaches while the closest games from TeamSkill-AllK-EV were presented to the baseline versions. The reasons these two were chosen is because all the TeamSkill approaches are intended to improve upon their respective baseline versions and that repeated testing had shown TeamSkill-AllK-EV to be the best performing approach for many subsets of the overall MLG Halo 3 dataset.

The advantage one team has over the other was computed in the following manner. Given two teams, t_1 and t_2, the prior probability of t_1 defeating t_2 is a straightforward derivation from the negative CDF at 0 of the distribution describing the difference

between two independent, normally-distributed random variables:

$$\begin{aligned} P(t_1 > t_2) = & \quad 1 - F(0; \mu_1 - \mu_2, \sigma_1^2 + \sigma_2^2) \\ = & \quad 1 - \frac{1}{2}(1 + erf(\frac{0 - (\mu_1 - \mu_2)}{\sqrt{2\sqrt{(\sigma_1^2 + \sigma_2^2)}^2}})) \\ = & \quad \frac{1}{2}(1 - erf(\frac{\mu_2 - \mu_1}{\sqrt{2(\sigma_1^2 + \sigma_2^2)}})) \end{aligned} \tag{3.5.5}$$

A final note: the default values used during the evaluation of Elo ($\alpha = 0.07$, $\beta = 193.4364$, $\mu_0 = 1500$, $\sigma_0^2 = \beta^2$), Glicko ($q = log(10)/400$, $\mu_0 = 1500$, $\sigma_0^2 = 100^2$), and TrueSkill ($\epsilon = 0.5$, $\mu_0 = 25$, $\sigma_0^2 = (\mu_0/3)^2$, $\beta = \sigma_0^2/2$) correspond to the defaults outlined in [1] and [19]. Additionally, for Glicko, a rating period of one game was assigned due to the continuity of game history over the course of 2008 and 2009, as well as to approximate an "apples to apples" comparison with respect to Elo and TrueSkill. In the interest of space, a subset of the 3,780 total evaluations corresponding to the "complete" cases are presented in this section, but a complete list of all the "long" and "recent" results are given in Appendix C.

3.6 Discussion

The results in figures 3.4, 3.5, and 3.6 show that in general, Glicko and TrueSkill benefit from the incorporation of team chemistry components and tend to improve the prediction accuracy overall in comparison to the baseline versions. The TeamSkill-AllK and TeamSkill-AllK-EV approaches - TeamSkill-AllK-EV in particular - outperform baseline in nearly all test cases. TeamSkill-AllK-LS, on the other hand, shows no similar performance gain, nor do any of the TeamSkill-K versions in the range $1 < k \leq 4$.

The performance of the TeamSkill-K cases for $1 < k \leq 4$ is, at first blush, disappointing. However, it is important to understand the context in which they're being evaluated. As *standalone* estimators of skill, they are relatively poor, with the occasional exception. It is also apparent that the higher k is, the worse TeamSkill-K's performance is. This is particularly evident for the results shown in figure 3.5, which is arguably the "cleanest" dataset as it consists of tournament games alone (online scrimmages are practice games, after all, and not always played with the same level of intensity). The

reason for the negative correlation between performance and k is due to the decreasing amounts of available game history for subgroups as k increases. With larger groups, every team change brings with it the need to "burn in" the new subgroup ratings, resulting in a period of time in which the overall team rating is unreliable. This issue plagues TeamSkill-AllK-LS as well, given its preference for larger subgroups.

Taken together, the results for TeamSkill-K and TeamSkill-AllK-LS suggest that subgroup-level ratings for $k > 1$ alone are insufficient for accurately assessing the strength of a team - ratings corresponding with individual players must be incorporated as well. The performance of these two approaches are perfect examples of the problems outlined in section 3.1.1.

Although the results for TeamSkill-AllK and TeamSkill-AllK-EV with Glicko or TrueSkill as the base learner are encouraging, no similarly positive effect is observed for Elo overall, with the exception of TeamSkill-AllK-EV's performance using Elo approaching that of baseline. In fact, the accuracy for *all* non-baseline approaches using Elo is, at best, equal to the baseline case. Interestingly, Elo still performs well in the baseline case, occasionally outperforming Glicko and TrueSkill. Considering Elo was developed in the mid-1950's, that it still competes with state-of-the-art approaches is itself an impressive result.

As to the source of Glicko and TrueSkill's improved overall performance, further inspection (figures 3.7, 3.8, and 3.9) reveals significant performance increases (with respect to baseline) in close games. At times, the margin of difference is as much as 8.6%, and in all close game scenarios with Glicko as the base learner, the differences relative to the baseline are significant ($p = 0.00799$ for tournament and scrimmage games, $p = 0.017$ for tournament games, and $p = 0.00886$ for scrimmage games). The difference is also significant for TrueSkill when considering all tournament and scrimmage games ($p = 0.01517$). It can also be seen that over time, this margin tends to widen. These results indicate that the subgroup-level ratings have the effect of better distinguishing which team has the true advantage in close match-ups, a key finding well-aligned with prior research [27], [28], and that the utility of team chemistry information in predicting the outcomes of such games grows as more subgroup game history is observed.

This is evident when reviewing the histograms of the final skill ratings for TeamSkill-K for each possible subgroup size k using Glicko as the base learner (see Figure 3.10).

A comprehensive set of skill rating histograms can be found in Appendix E. To account for the variance parameters of each skill assessment technique, 3σ was subtracted from each skill rating μ to produce a "conservative" skill estimate (according to the method outlined in [1]). While the majority of subgroup skill ratings are clustered around the default conservative skill rating for a particular choice of k, the results clearly show several subgroups of size $k = 2$ and $k = 3$ whose conservative skill estimates are far above normal, and higher than any summation of k skill ratings from $k = 1$. This is only possible when the performances of a subgroup far exceed that of any subgroup member's individual performance history, which we attribute to the team chemistry within their subgroup.

This helps us understand why Elo doesn't benefit from the inclusion of group-level ratings information. The reason stems from Elo's use of a constant variance and as such, Elo is not sensitive to the dynamics of a player's skill over time - all ratings are implicitly assumed to have equal reliability regardless of how much game history has been observed. For groups of players, this issue is compounded since the higher the k under consideration, the less prior game history can be drawn on to infer their collective skill. With the TeamSkill approaches, the net effect is that incorporating $(k > 1)$-level group ratings 'dilute' the overall team rating, resulting in a higher number of closer games since there is no provision for Elo's constant variance to differ depending on the size of the group under consideration.

Similarly, variance also accounts for much of the differences between Glicko and TrueSkill's performances. Both make use of player-level variances (and, thus, group-level variances using the TeamSkill approaches). However, TrueSkill also maintains a constant "performance" variance, β^2, across all players, which is applied just prior to computing the predicted ordering of teams during updates. β^2 is a user-provided parameter which, when increased, similarly increases the probability of TrueSkill believing teams will draw, discounting the potentially small differences between them in collective skill. As such, this "performance" variance has a similar 'dilution' effect as in Elo, but are less pronounced because TrueSkill also maintains player/subgroup-level variances. These results also highlight the critical role played by skill variance in estimating the skill of a group of players.

With the superior performance of TeamSkill-AllK and TeamSkill-AllK-EV using

Glicko and TrueSkill as base learners, the nagging question is whether or not these approaches could do better were the aggregation weights managed in a less naïve fashion. We will explore this topic further in the following section.

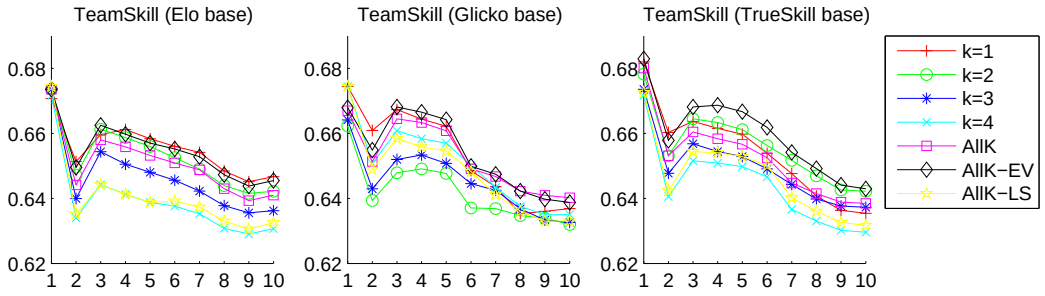


Figure 3.4: Prediction accuracy for both tournament and scrimmage/custom games using complete history.

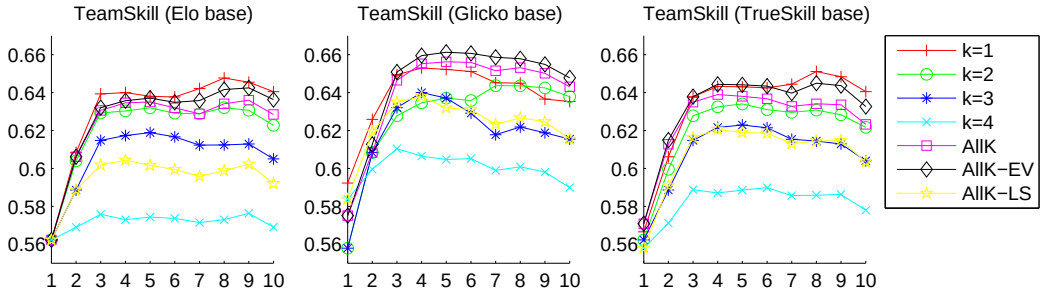


Figure 3.5: Prediction accuracy for tournament games using complete history.

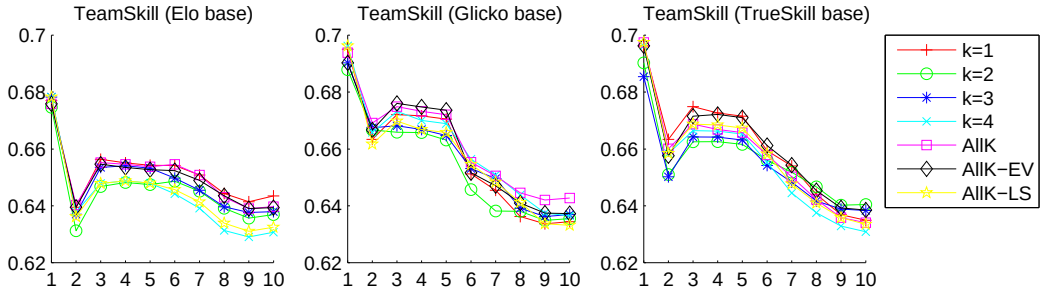


Figure 3.6: Prediction accuracy for scrimmage/custom games using complete history.

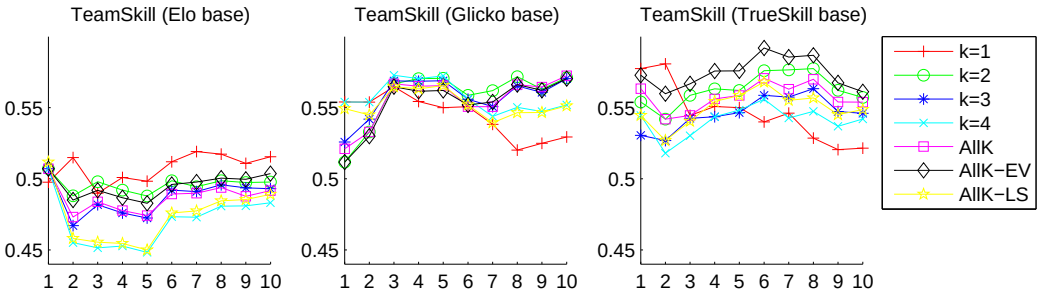


Figure 3.7: Prediction accuracy for both tournament and scrimmage/custom games using complete history, close games only.

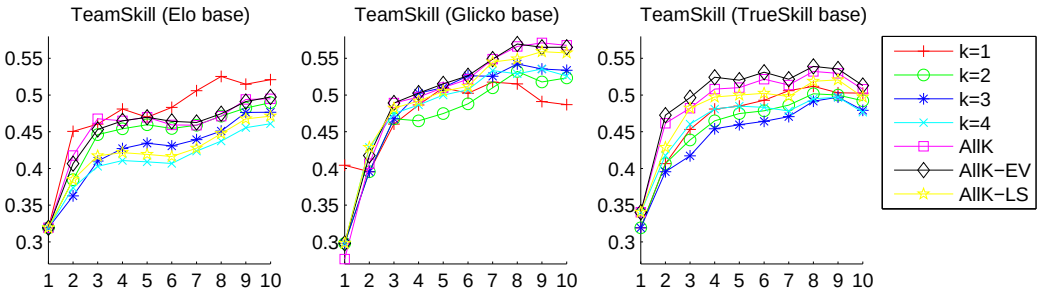


Figure 3.8: Prediction accuracy for tournament games using complete history, close games only.

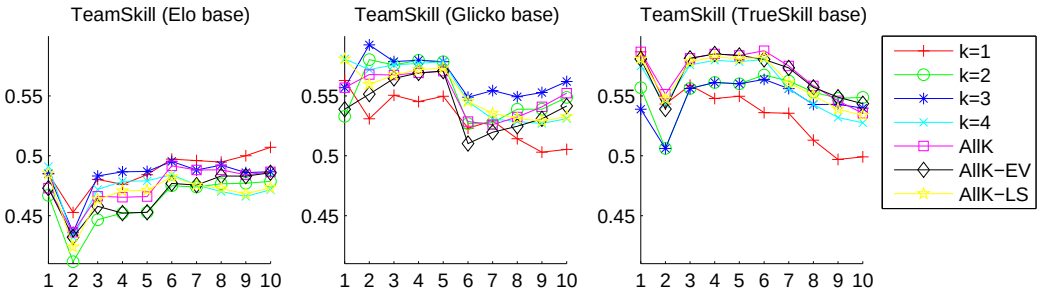


Figure 3.9: Prediction accuracy for scrimmage/custom games using complete history, close games only.

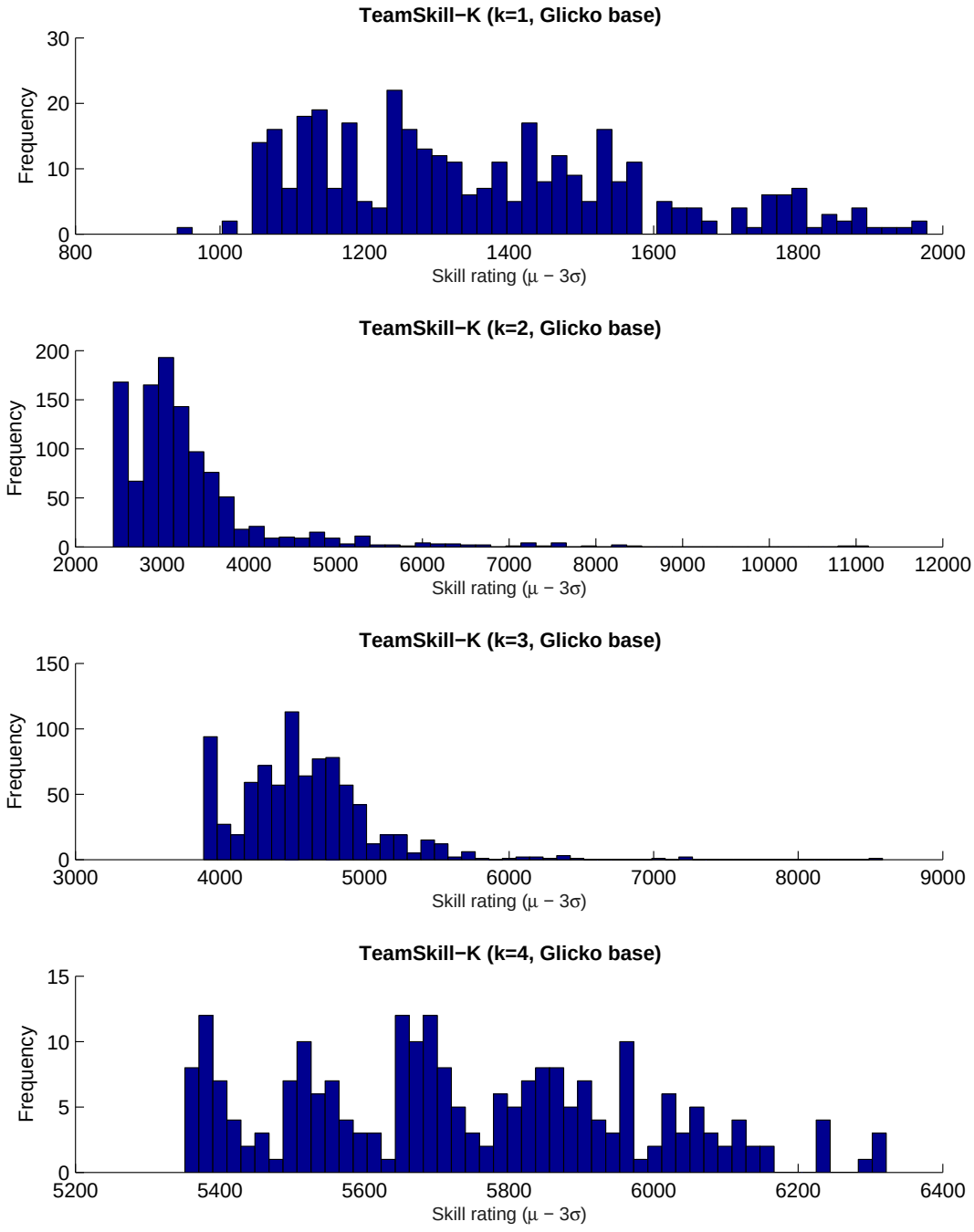


Figure 3.10: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with Glicko ($\mu_0 = 1500, \sigma_0^2 = 100^2$) as the base learner, tournament games only, and complete history.

Chapter 4

Optimizing sets of weights during the aggregation step

Though several of the TeamSkill approaches described in the previous section performed well relative to the baseline baseline learners, they are nonetheless still naïve in formulation, having no dynamic means of aggregating subgroup ratings together to arrive at an overall rating for a team. Because of this, these approaches are possibly suboptimal due to two key perceived weaknesses: the "group history problem" and the "dynamic feature space problem".

4.1 Perceived problems with the naïve approaches

4.1.1 The group history problem

Section 3.1.1 describes one of the perceived problems of naïve approaches like TeamSkill-K, which TeamSkill-AllK and TeamSkill-AllK-EV are meant to address. However, their method of addressing this issue is similarly naïve in that there is no dynamic weighting of the aggregate ratings corresponding to each of the k-sized subgroups, potentially leading to a less than ideal overall skill rating for a team. For any given game, the subgroups of a team will almost certainly have varying amounts of prior game history and, therefore, subgroup ratings of varying reliability. Though ameliorated by the dynamic variance parameters of the Glicko and TrueSkill base learners, it is worth exploring a dynamic

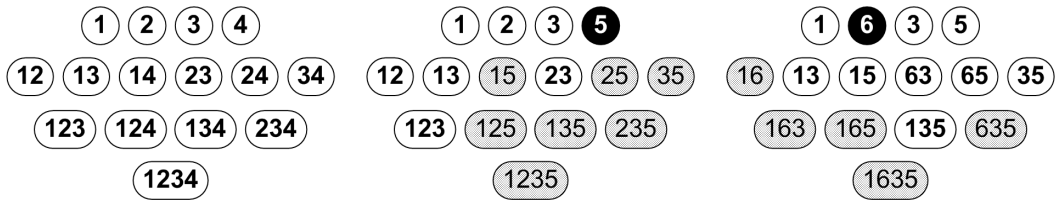


Figure 4.1: The dynamic feature space problem. This figure illustrates the group history available for a team of four players at three different time instances, proceeding chronologically from left to right. Unshaded cells indicate that history is available for a given group while lightly-shaded cells indicate that history is not available.

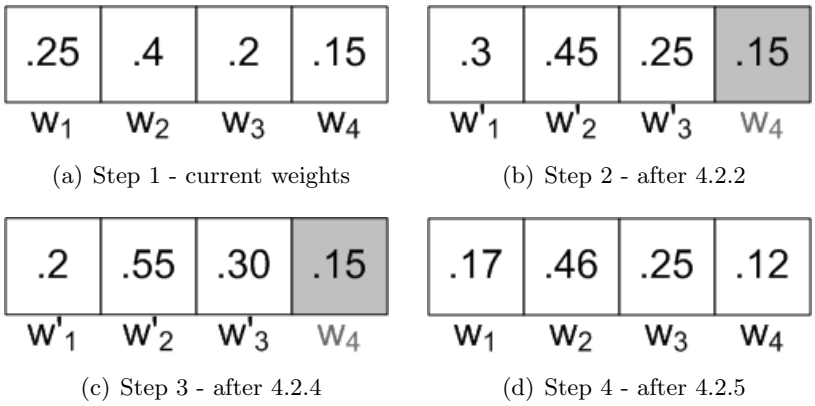
weighting scheme to see if further mitigation of this issue is warranted.

4.1.2 The dynamic feature space problem

When attempting to construct an overall team skill rating, one key challenge to overcome is the fact that the amount of subgroup history can vary over time. Consider figure 4.1: after the first game is played, history is available for all possible subgroups of players. Later, player 4 leaves the team and is replaced by player 5, who has never played with players 1, 2, or 3, leaving only a subset of history available and none for the team as a whole. Then in the final step, player 2 leaves and is replaced by player 6, who has played with player 3 and 5 before, but never both on the same team, resulting in yet another variant on the team's collective subgroup history. Thus the feature space is constantly expanding and contracting over time, making it difficult to know how best to combine the subgroup ratings together.

Let us use a weather prediction metaphor to help better understand this problem. Suppose we propose a new way of predicting the weather. Each morning, we invite a number of weather experts to a park and have them look at the clouds in the sky, take measurements of temperature, barometric pressure, etc., and ask each of them to predict what the weather will be like that day. Our prediction, we decide, will be the weighted majority opinion of the experts' opinions.

After several days of this new weather prediction process, we find that although the weather experts have largely overlapping skill sets, some possess unique skills as well. Several experts discover that they work well with each other, exploiting their unique skills in a complementary fashion to better predict the weather each day. Naturally,



(a) Step 1 - current weights

(b) Step 2 - after 4.2.2

(c) Step 3 - after 4.2.4

(d) Step 4 - after 4.2.5

Figure 4.2: Example of TeamSkill-AllK-EV-OL1 weight update process for $K' = 3$ where $P_1(i > j) < .5$, $P_2(i > j) > .5$, and $P_3(i > j) > .5$

we note which individuals and groups are making the most accurate predictions, giving more weight to those who do well and less weight to those who do not.

However, there is a twist to this otherwise straightforward weighted majority prediction problem: not everyone shows up each morning. Some show up a few times and then stop, others join later on, and some are there every day. As a result, if we think of each expert and any groups they form as classifiers, or "features", for our prediction task, that feature space is expanding and contracting over time.

This is the dynamic feature space problem. The prediction task is a function of "who shows up", or which ratings we have prior game history for, and per the previous section, we may have very different amounts of game history subgroup to subgroup. This makes it challenging to find a mechanism for maintaining aggregation weights when different subgroups will have observed different (and overlapping, to some extent) game histories. How do we combine these subgroup ratings together optimally?

4.2 TeamSkill-AllK-EV-OL1

Our first attempt at tackling this issue is to maintain a constantly-updated weight vector w_k for each aggregated subgroup skill rating and resize $\mathbf{w}$ according to the size of the largest subgroup with available history across both competing teams, K'. In doing so, we redistribute evenly the sum of the unused weights ($k > K'$) to the rest of the weight

vector, retaining a semblance of the original weight distribution while allowing us to estimate each team's skill using the available subgroup ratings ($k \leq K'$). Here, the updates for w_k are proportional to the prior probability of team i defeating some team j according to TeamSkill-K [35], $P_k(i > j) > 0.5$, as given in the set of equations 4.2 below.

$$1 \leq \beta \leq \infty, w_k^0 = \frac{1}{K},$$

$$K' = \min\left(\max_{k \leq K}\left(|h_i(k)| > 0\right), \max_{k \leq K}\left(|h_j(k)| > 0\right)\right)$$

$$u = \frac{1}{K'} \sum_{k > K'} w_k^t \tag{4.2.1}$$

$$w_{(k \leq K')}'^t = w_{(k \leq K')}^t + u \tag{4.2.2}$$

$$s_i^* = \sum_{k=1}^{K'} w_k'^t E[h_i(k)] \tag{4.2.3}$$

$$w_{(k \leq K')}^{t+1} = w_{(k \leq K')}^t \beta^{\frac{1}{2} + P_k(i > j)} \tag{4.2.4}$$

$$w_k^{t+1} = \frac{w_k^{t+1}}{\sum_{l=1}^{K} w_l^{t+1}} \tag{4.2.5}$$

Figure 4.2 illustrates the weight redistribution and update process for teams of 4 players. With $K' = 3$, only w_1, w_2, and w_3 can be used for estimating the overall skill of each team, and so the weight w_4 is evenly redistributed to the other weights (figure 4.2.2). Each team's skill is estimated in 4.2.3. According to TeamSkill-K, $P_1(i > j) < .5$ and so w_1 is decreased, while $P_2(i > j) > .5$ and $P_3(i > j) > .5$, increasing the weights of w_2 and w_3, respectively (figure 4.2.4). Finally, $\mathbf{w}$ is rebuilt into a distribution in 4.2.5, completing the update process.

From figure 4.2, it is apparent that subgroups ratings which contribute positively to the prediction of which team will win have their weights increased, but it is also apparent despite w_4 not being used in the estimation, its weight is nonetheless decreased. Less obvious is the observation that *most* base learners will do better than random i.e., $P_k(i > j) > 0.5$, in most cases, and so the weights for the ratings of the most frequently used subgroup sizes will tend to increase over time. As such, OL1 will eventually converge to the baseline classifier, or TeamSkill-K where $k = 1$, because individual

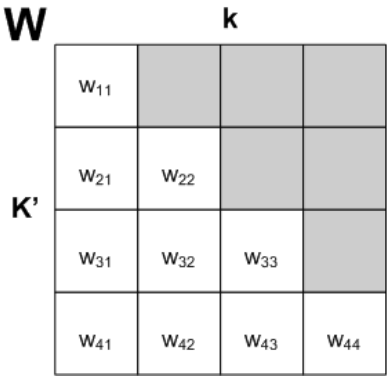


Figure 4.3: Weight matrix of size KxK for TeamSkill-AllK-EV-OL2.

player history is available at least as often as subgroups comprised of said players, resulting in $w_1 \to 1$. This is TeamSkill-AllK-EV-OL1's primary weakness.

4.3 TeamSkill-AllK-EV-OL2

As a means of overcoming TeamSkill-AllK-EV-OL1's perceived shortcoming, a KxK matrix of weights (versus a single weight vector) was introduced, $\mathbf{W}$ (see figure 4.3). The weight matrix is essentially a vector of vectors, but because each row corresponds to a use case for each possible value of K', only the lower triangular of the matrix is used. Maintaining the weights in this way makes it impossible for one w_k, particularly $k = 1$, to dominate the weighting simply because it has the highest subgroup frequency.

The formal description of this approach, TeamSkill-AllK-EV-OL2, is given below. Since we are not resizing a single weight vector and instead choosing the row vector of size K' from the matrix $\mathbf{W}$, the approach is greatly simplified relative to Ol1.

$$s_i^* = \sum_{k=1}^{K'} w_{(K',k)}^t E[h_i(k)] \tag{4.3.1}$$

$$w_{(K',k\leq K')}^{t+1} = w_{(K',k\leq K')}^t \beta^{\frac{1}{2}+P_k(i>j)} \tag{4.3.2}$$

$$w_{(K',k)}^{t+1} = \frac{w_{(K',k)}^{t+1}}{\sum_{l=1}^{K'} w_{(K',l)}^{t+1}} \tag{4.3.3}$$

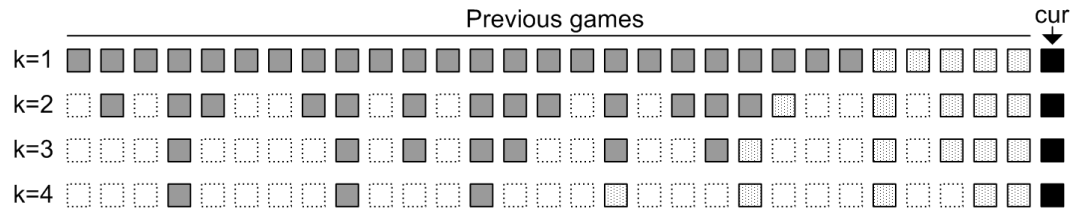


Figure 4.4: Example of window process for TeamSkill-AllK-EV-OL3 ($d = 5$ games). Shaded boxes (light or dark) indicate games for which ratings were available for each subgroup of size k. Lightly-shaded boxes are games occurring within the window of d most recent games for a given k. Boxes shaded all in black refer to the current game we are trying to predict the outcome of using weights determined by the loss $L_{d,k}$ suffered by TeamSkill-K over the window d for some k.

Though this approach would appear to address the major issues raised by prior TeamSkill iterations, OL2 still leaves one problem largely intact: the group history problem. Despite splitting the weights into K different vectors, the weights of less common subgroups are still based on the corresponding predictive accuracy of TeamSkill-K, which are known to perform rather poorly as standalone classifiers, at least until enough game history has accumulated. As such, the weight vectors again tend to bias toward the $w_{K',1}$ weights when estimating the overall skill of a team.

4.4 TeamSkill-AllK-EV-OL3

To address both OL1 and OL2's weaknesses, a third variant - TeamSkill-AllK-EV-OL3 - was developed. OL3 is essentially a version of OL1 which attempts to address OL1's deficiency by maintaining a window of the d most recent games in which k-sized subgroup history was available, updating w_k according to the loss $L_{d,k}$ suffered by TeamSkill-K over the window d for some k. As such, the prevalence of individual player history (i.e., w_1) is, in some sense, factored out of the updates to w_k since there are no more than d games available for each k at any given time.

$$s_i^* = \sum_{k=1}^{K'} w_k'^t E[h_i(k)] \tag{4.4.1}$$

$$w_{(k \leq K')}^{t+1} = w_{(k \leq K')}^{t} \beta^{\frac{1}{2} + (d - L_{d,k})/d} \tag{4.4.2}$$

$$w_k^{t+1} = \frac{w_k^{t+1}}{\sum_{l=1}^{K} w_l^{t+1}} \tag{4.4.3}$$

Figure 4.4 demonstrates how the window process works. Here, the 5 most recent games are used for *each* aggregate subgroup rating of size k, as opposed to a *global* window of 5 games, which would otherwise contain only 3 games for w_4. This approach places each aggregate subgroup rating on a relatively similar footing with respect to each other from a game history perspective, and from the weighting scheme, ascribes more weight to only those subgroup ratings which are more accurately predicting the outcomes of games (i.e., $(d - L_{d,k})/d \to 1$), eliminating the drawback of OL1 and, to a great extent, the group history issue of OL2 as well. That is, because the subgroup ratings must "prove" themselves in order for their weight to be increased, the problem of not having much group history for newly formed teams and subgroups is largely mitigated.

4.5 Experiments

As in section 3.5, the prediction accuracy of OL1, OL2, and OL3 were evaluated with the Elo, Glicko, and TrueSkill base learners using several different subsets of the MLG Halo 3 dataset [36]:

- Games played in tournaments only, scrimmage games only, and both tournament and scrimmage games.
- All of the games, or just those games considered "close" (i.e., prior probability of one team winning close to 50%).

For comparison, we include results from the previous TeamSkill approaches as well. Because we are interested in the overall performance of OL1, OL2, and OL3 relative to the previous TeamSkill approaches, as well as the baseline, the approaches were

Table 4.1: Overall prediction accuracy for all test cases. **Bold cells** = highest accuracy; ***bolded/italicized*** = 2nd-highest accuracy.

Learner	Data	Close?	k=1	k=2	k=3	k=4	AllK	AlKEV	AlKLS	OL1	OL2	OL3
Elo	Both	N	***0.645***	0.642	0.636	0.631	0.642	***0.645***	0.633	***0.645***	***0.645***	**0.646**
		Y	**0.512**	0.494	0.497	0.485	0.493	0.500	0.489	0.495	0.495	***0.502***
	Tourn.	N	**0.639**	0.626	0.607	0.571	0.628	0.635	0.592	**0.639**	**0.639**	0.633
		Y	0.518	0.497	0.482	0.464	0.500	0.510	0.474	***0.531***	**0.536**	0.510
	Scrim.	N	**0.643**	0.639	0.639	0.631	0.642	0.640	0.633	**0.643**	**0.643**	0.640
		Y	**0.503**	0.487	0.492	0.476	0.496	0.488	0.476	***0.499***	0.498	0.487
Glicko	Both	N	0.636	0.630	0.632	0.635	**0.640**	**0.640**	0.633	0.637	0.637	**0.640**
		Y	0.522	0.564	0.562	0.547	0.569	***0.570***	0.548	0.524	0.552	**0.571**
	Tourn.	N	0.638	0.637	0.616	0.588	0.644	**0.647**	0.613	0.637	0.637	**0.647**
		Y	0.484	0.529	0.531	0.523	**0.576**	***0.570***	0.557	0.526	0.560	***0.570***
	Scrim.	N	0.631	0.635	***0.637***	***0.637***	**0.643**	***0.637***	0.634	0.635	0.636	***0.637***
		Y	0.496	0.559	**0.565**	0.522	***0.562***	0.551	0.524	0.531	0.551	0.551
TrueSkill	Both	N	0.635	0.641	0.636	0.630	0.638	***0.642***	0.632	0.635	0.636	**0.643**
		Y	0.516	0.555	0.542	0.542	0.552	***0.560***	0.548	0.536	0.544	**0.562**
	Tourn.	N	0.640	0.626	0.601	0.576	0.626	0.636	0.601	***0.641***	**0.644**	0.634
		Y	0.500	0.497	0.479	0.474	0.508	0.510	0.495	***0.531***	**0.547**	0.508
	Scrim.	N	0.636	**0.642**	***0.639***	0.632	0.636	0.638	0.634	0.636	0.637	0.637
		Y	0.504	**0.550**	0.542	0.530	0.541	0.540	0.533	0.548	**0.550**	0.543

evaluated using "complete" data history described in 3.5. Additionally, the defaults of OL1/OL2 ($\beta = 1.1$) and OL3 ($d = 20$) were assumed.

The results of the evaluation are shown in table 4.1. It is readily apparent that OL3 performs essentially the same as AllK-EV, and only slightly better for the set of "close" scrimmages and tournament games. OL1 and OL2, though improving upon the baseline in all but one case each (the set of tournament games), do not achieve the performance of AllK-EV.

4.6 Discussion

Overall, the evaluation results for the dynamic weight management approaches OL1, OL2, and OL3 are disappointing.

OL1 and Ol2 fail to achieve parity with the best "naïve" approaches of AllK and AllK-EV, with OL1 falling especially short, providing at best an example of how the group history and dynamic feature space problems are difficult to solve. In OL1, subgroup rating weights for $k > 1$ are driven nearly to 0 due to insufficient game history, and only slowly begin to recover after many games are seen, but not quick enough to additionally address the issue of new subgroups being formed as players change teams.

In OL2, the decreased performance relative to OL3 is much less pronounced, indicating that much of OL1's deficit is accounted for by the aforementioned $k > 1$ subgroup weight problem, and further that OL2 suffers from a version of this issue as well, albeit to a lesser extent, by failing to more heavily weight $k > 1$ subgroup ratings and, thus, incorporate the improved predictive power afforded by subgroup ratings with more game history later on.

The performance of OL3 suggests that AllK and AllK-EV's naïve weight aggregation approaches are, in fact, strong choices for the general case. Because the loss framework of OL3 is meant to capture the aggregate subgroup ratings which are performing the best in the most recent d games, the results further underscore the importance of not discounting subgroup ratings after enough games have been observed to make use of subgroup-level skill driven, in part, by team chemistry.

Taken together, the results appear to show that having a dynamic means of maintaining aggregation weights realizes little benefit at best, and quickly degrades predictive performance at worst. That said, there is yet more unexplored territory. The incorporation of game-specific data may provide an alternative option for improving predictive performance, so long as it is done in a generalized framework flexible enough to not stray far from the "general case".

Chapter 5

Enhancing the model using game-specific data

OL1, OL2, and OL3 - like the other TeamSkill approaches - only use data available in all team-based games, namely the players, their team associations, game outcome history, etc 1.2.2. One natural question to ask is how well we could do were game-*specific* data included during the step in which the label of the winning team is predicted. Though not *ideal* from the perspective of building a generalizable skill assessment framework, we introduce two methods - TeamSkill-AllK-EVGen (EVGen) and TeamSkill-AllK-EVMixed (EVMixed) - which are are robust enough to accommodate a carefully-chosen set of game-specific performance metrics for any game.

5.1 TeamSkill-AllK-EVGen

We begin with EVGen, which constructs a feature set $\mathbf{x}_t$ at some time instance t from a combination of EV's predicted label $\{+1, -1\}$ of the winning team, $\hat{EV}_t$ [1] , and a set of n game-specific metrics $\mathbf{m}$, the result of which is then used to learn a weight vector $\mathbf{w}$ for some choice of an online classification framework and tested on unseen games.

The game-specific metrics may be of any numerical variety. For Halo 3, a team-based first person shooter, several logical metrics are available, such as kill/death ratio and assist/death ratio (an assist is given to a player when they do more than half of the

[1] $sign(P(EV_{it} > EV_{jt}) - 0.5) \equiv \hat{EV}_t$

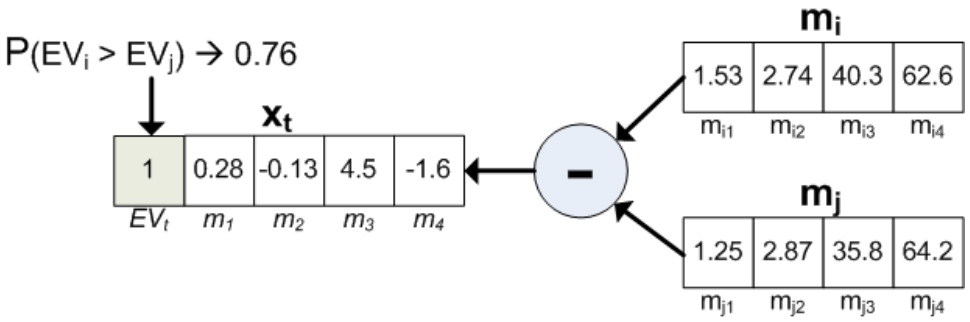


Figure 5.1: Example of process for constructing feature set $\mathbf{x}_t$, where m_i and m_j are collections of game-specific metrics for teams i and j, respectively.

damage to a player who is eventually killed by another player), and act as rough measures of a team's in-game efficiency since players respawn after each death throughout the duration of a game. Keeping the opposing team on constant respawn is crucial for fulfilling game type objectives, such as establishing map position or capturing flags.

After compiling these metrics for each team, we take the difference between them for use in $\mathbf{x}_t$, adding in the predicted label for TeamSkill-AllK-EV, $\hat{EV}_t$, as the final feature (see Figure 5.1 for an example). AllK-EV was chosen because of its consistently good performance in previous evaluations as well as results from preliminary testing for this work, drawing from the pool of all previous approaches (including OL1, OL2, and OL3). The resulting feature vector $\mathbf{x}_t$ can be formalized as follows:

$$\mathbf{x}_t = (\hat{EV}_t, m_1, m_2, ..., m_n) \tag{5.1.1}$$

Having constructed the feature set $\mathbf{x}_t$, we use a more traditional online classification framework to predict the label of the winning team $\hat{y}_t$, such as the perceptron [37], online Passive-Aggressive algorithms [38], or Confidence-Weighted learning [39]. With the exception of the Confidence-Weighted learning, in which $\boldsymbol{\mu}_t$ is substituted for $\mathbf{w}_t$, unseen instances are assigned predicted labels according to the sign of the dot product of the feature vector $\mathbf{x}_t$ and their corresponding weights $\mathbf{w}_t$:

$$\hat{y}_t = sign(\mathbf{w}_t \cdot \mathbf{x}_t) \tag{5.1.2}$$

Once the label has been predicted, the weight vector $\mathbf{w}_t$ is then updated according to the chosen online classification framework. As in all such classifiers, the core task

is to iteratively learn $\mathbf{w}_t$ such that the expected loss on unseen instances is minimized. The choices of loss function and minimization criteria are the key traits differentiating online learning algorithms from each other.

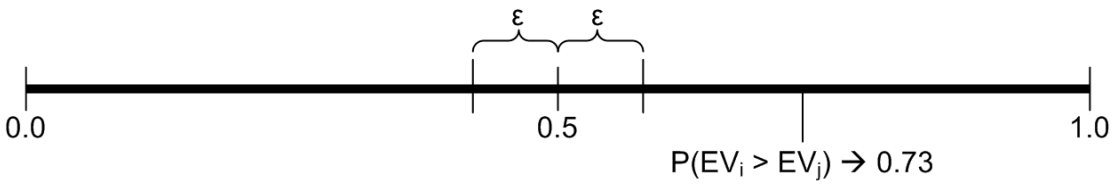
Because this approach predicts the outcome of each game using the entire feature vector $\mathbf{x}_t$, the underlying assumption is that game-specific metrics are relevant in all cases. It may not always be so. Consider the case of the major and minor leagues of baseball: players may have similar batting averages, but no one would claim that the minor leagues was more difficult than the majors, and so minor league batting average may be of little relevance to the task of predicting how a player would perform against much stronger competition. As such, when looking across the set of *all* games (such as the semi-imaginary case where minor league teams might compete against major league teams), the unimportance of a game-specific feature like batting average *in a global sense* could outweigh the potentially positive effect of utilizing it in the cases where it might be expected to make a difference. A new approach is needed which can address this concern.

5.2 TeamSkill-AllK-EVMixed

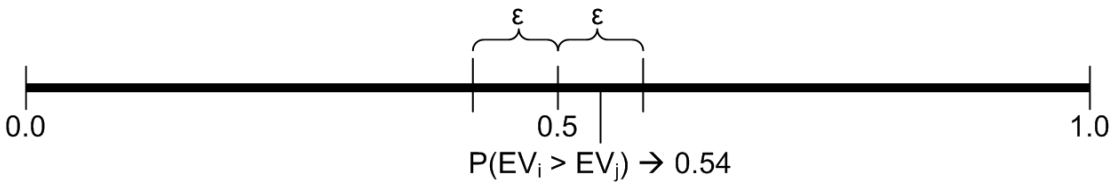
Our proposed solution is simple. If the skill ratings from TeamSkill-AllK-EV suggest that two teams are more or less evenly-matched (i.e., a team's prior probability of winning according to AllK-EV, $P(EV_{it} > EV_{jt})$, is close to 0.5), then we might expect that game-specific metrics would be more meaningful since the strength of each team's prior opposition has been accounted for. Alternatively, if one team is far enough in skill from the other team, then we ignore the game-specific features and predict the label of the winning team based on AllK-EV alone. In this alternative case, our intuition is that game-specific metrics are useless - or harmful - with respect to the task of predicting the label of the winning team when their skills diverge significantly. Accordingly, we introduce a parameter ϵ to act as the threshold between prediction strategies:

$$\hat{y}_t = \begin{cases} sign(\mathbf{w}_t \cdot \mathbf{x}_t) & \text{if } |P(EV_{it} > EV_{jt}) - 0.5| < \epsilon \\ \hat{EV}_t & \text{otherwise} \end{cases} \tag{5.2.1}$$

Two examples of how EVMixed handles these cases are provided in figures 5.2(a)



(a) Case 1 - Teams i and j are not evenly-matched



(b) Case 2 - Teams i and j are evenly-matched

Figure 5.2: How TeamSkill-AllK-EVMixed works. In the first case, $P(EV_i > EV_j) = 0.73$, so EVMixed predicts the outcome of the game based on the label $\hat{EV}$ which, in this case, is team i. In the second case, $P(EV_i > EV_j) = 0.54$, and since from this example $|0.54 - 0.5| < \epsilon$, EVMixed predicts the outcome of the game according to $sign(\mathbf{w} \cdot \mathbf{x})$.

and 5.2(b).

5.3 Experiments

The evaluation of EVGen and EVMixed was conducted using subsets of the MLG Halo 3 dataset and the same experimental framework as in Section 4.5, and tested with Elo, Glicko, and TrueSkill as base learners:

- Games played in tournaments only, scrimmage games only, and both tournament and scrimmage games.
- All of the games, or just those games considered "close".

Because Halo 3 is a team-based first-person shooter with players respawning infinitely until each game is concluded, a team's in-game effort and efficiency can be inferred from a set of rough proxy measures based on the rate at which a given player or team can kill opposing players (kills) or aid in killing them (assists) while minimizing their own deaths. From table 5.1 there are 4 variations on 4 core metrics, for a total

Table 5.1: List of game-specific features used for evaluating the performance of EVGen and EVMixed.

Variable Name
Kill/death ratio (cumulative, $k = K$)
Assist/death ratio (cumulative, $k = K$)
Kills per game (cumulative, $k = K$)
Assists per game (cumulative, $k = K$)
Kill/death ratio (cumulative, $k = 1$)
Assist/death ratio (cumulative, $k = 1$)
Kills per game (cumulative, $k = 1$)
Assists per game (cumulative, $k = 1$)
Kill/death ratio (last d games, $k = K$)
Assist/death ratio (last d games, $k = K$)
Kills per game (last d games, $k = K$)
Assists per game (last d games, $k = K$)
Kill/death ratio (last d games, $k = 1$)
Assist/death ratio (last d games, $k = 1$)
Kills per game (last d games, $k = 1$)
Assists per game (last d games, $k = 1$)

of 16 features. "Cumulative" considers the entire history of a team or set of players prior to the current game compared to the more recenty history of "last d games". For $k = K$, we consider the performance of the subgroup corresponding to the entire team for a given set of history and, for $k = 1$, only the subgroups coterminous with individual player histories are used.

For EVGen and EVMixed, the threshold $\epsilon = 0.03$. This setting was chosen based on extensive preliminary testing indicating that 0.03 - games in which the likelihood $P(EV_{it} > EV_{jt})$ of team i defeating team j was between 47% and 53% - produced the highest observed prediction accuracy on the MLG Halo 3 dataset. Additionally, we selected the Passive-Aggressive II algorithm [38] as the online classifier based on similar preliminary testing, a fuller description of which can be found later in Section 5.3.3. The default parameter settings of $\alpha = 0.1$, $C = 0.001$, and $\eta = 0.9$ are unchanged from the evaluations conducted by the authors [38]. Finally, and also based on early evaluation results, the window parameter d was set to 10 games.

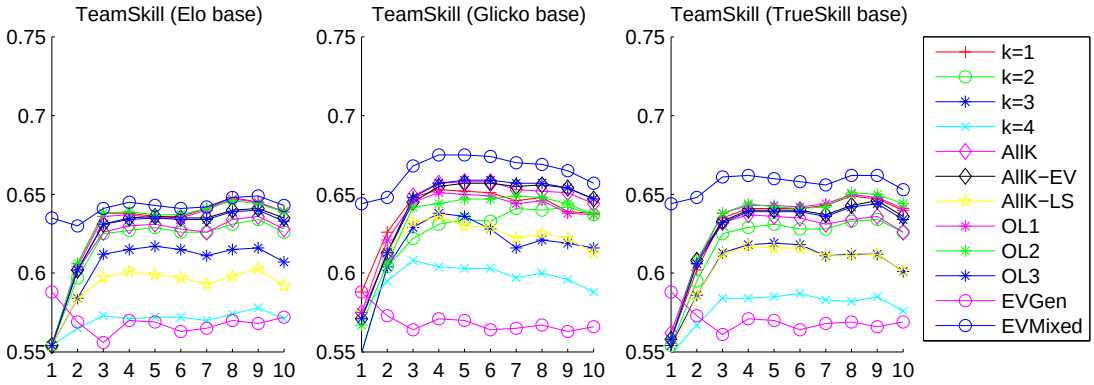
Table 5.2: Overall prediction accuracy for all test cases. **Bold cells** = highest accuracy; ***bolded/italicized*** = 2nd-highest accuracy.

Learner	Data	Close?	k=1	k=2	k=3	k=4	AllK	AlKEV	AlKLS	OL	OL2	OL3	EVGen	EVMxd
Elo	Both	N	0.645	0.642	0.636	0.631	0.642	0.645	0.633	0.645	0.645	***0.646***	0.574	**0.647**
		Y	0.512	0.494	0.497	0.485	0.493	0.5	0.489	0.495	0.495	0.502	**0.523**	***0.521***
	Tourn.	N	***0.639***	0.626	0.607	0.571	0.628	0.635	0.592	***0.639***	***0.639***	0.633	0.572	**0.643**
		Y	0.518	0.497	0.482	0.464	0.5	0.51	0.474	0.531	0.536	0.51	**0.549**	***0.544***
	Scrim.	N	***0.643***	0.639	0.639	0.631	0.642	0.64	0.633	***0.643***	***0.643***	0.64	0.583	**0.644**
		Y	0.503	0.487	0.492	0.476	0.496	0.488	0.476	0.499	0.498	0.487	**0.529**	***0.512***
Glicko	Both	N	0.636	0.63	0.632	0.635	***0.64***	***0.64***	0.633	0.637	0.637	***0.64***	0.581	**0.641**
		Y	0.522	0.564	0.562	0.547	0.569	0.57	0.548	0.524	0.552	***0.571***	0.528	**0.573**
	Tourn.	N	0.638	0.637	0.616	0.588	0.644	***0.647***	0.613	0.637	0.637	***0.647***	0.566	**0.657**
		Y	0.484	0.529	0.531	0.523	***0.576***	0.57	0.557	0.526	0.56	0.57	0.518	**0.62**
	Scrim.	N	0.631	0.635	0.637	0.637	**0.643**	0.637	0.634	0.635	0.636	0.637	0.582	***0.638***
		Y	0.496	0.559	***0.565***	0.522	***0.562***	0.551	0.524	0.531	0.551	0.551	0.525	0.554
TrueSkill	Both	N	0.635	0.641	0.636	0.63	0.638	0.642	0.632	0.635	0.636	**0.643**	0.572	**0.643**
		Y	0.516	0.555	0.542	0.542	0.552	0.56	0.548	0.536	0.544	**0.562**	0.522	***0.561***
	Tourn.	N	0.64	0.626	0.601	0.576	0.626	0.636	0.601	0.641	**0.644**	0.634	0.569	**0.653**
		Y	0.5	0.497	0.479	0.474	0.508	0.51	0.495	0.531	***0.547***	0.508	0.542	**0.573**
	Scrim.	N	0.636	**0.642**	0.639	0.632	0.636	0.638	0.634	0.636	0.637	0.637	0.581	***0.64***
		Y	0.504	**0.55**	0.542	0.53	0.541	0.54	0.533	0.548	**0.55**	0.543	0.522	0.542

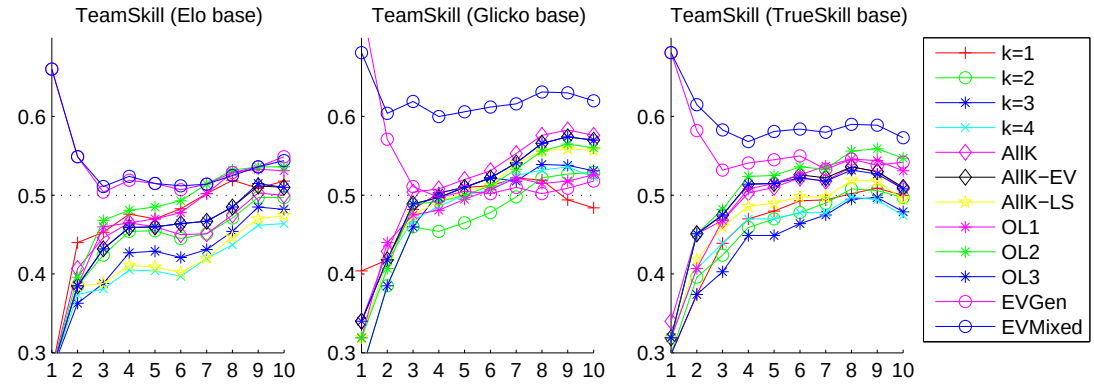
5.3.1 Overall accuracy

From the results in table 5.2, it is readily apparent that EVMixed performs the best overall, and in the widest array of evaluation conditions. EVMixed outperforms both the baseline and previous "best" approach AllK-EV in every case, has the best performance in 10 of the 18 test cases, and 16 of 18 in which it was at least second best, all of which are a testament to its EVMixed's consistency. The most improvement is seen in close games with Glicko as the base learner, all of which are statistically-significant at no worse than $p = 0.577$ in any case. For tournament data especially, in which EVMixed outperforms the baseline by 13.6%, the difference is statistically-significant at $p = 0.00015$. Strong results are also seen for EVMixed in close games with TrueSkill as the base learner, with performance exceeding the baseline between 3.8% to 7.3% in prediction accuracy, and are statistically-significant improvements in all but one case (scrimmage data only).

In contrast, EVGen's overall performance is disappointing, between $5 - 7\%$ lower over all games, and exceeding EVMixed's results only in 3 of the "close" game test cases. In the other "close" game cases, EVGen, while slightly improved over the baseline, is between $2 - 10\%$ worse than EVMixed.



(a) Prediction accuracy over time for tournament games.



(b) Prediction accuracy over time for tournament games, close games only.

5.3.2 Results over time

Next we explore how these approaches perform over time by predicting the outcomes of games occurring prior to 10 tournaments which took place during 2008 and 2009, using tournament data only in order to isolate conditions in which we expect teamwork to be strongest. In tournament games, players and teams are competing at the highest level and have a large financial incentive to perform well, and as such, this subset of the overall dataset has consistently produced the best results for the suite of TeamSkill approaches.

From figures 5.3(a) and 5.3(b), EVMixed's superior performance is easily observable. Of particular note, however, is how well EVMixed does when little history is available,

Table 5.3: Comparison of prediction accuracy by online classification framework using Glicko as the base learner. **Bold cells** = highest accuracy; ***bolded/italicized*** = 2nd-highest accuracy.

		EVGen			EVMixed		
Data	Close?	Perceptron	PA-II	CW-diag	Perceptron	PA-II	CW-diag
Both	N	0.575	***0.581***	**0.584**	**0.641**	**0.641**	**0.641**
	Y	***0.514***	**0.528**	**0.528**	**0.573**	**0.573**	**0.573**
Tourn.	N	0.543	**0.566**	***0.564***	***0.655***	**0.657**	**0.657**
	Y	0.474	**0.518**	***0.51***	0.609	**0.62**	***0.617***
Scrim.	N	0.575	***0.582***	**0.586**	**0.638**	**0.638**	***0.637***
	Y	***0.515***	**0.525**	0.512	**0.556**	***0.554***	0.551

having a roughly 64% accuracy just prior to the first tournament for all three base learner cases. EVGen exhibits similar behavior in the beginning, but quickly drops off as more game history is seen, never recovering to the performance it had early on.

For close games, both EVGen and EVMixed show strong results, eventually tapering off and approaching the other competing methods as more game history is observed. EVMixed, though, always maintains a noticeable gap of improvement relative to the other approaches, particularly for Glicko and, to a lesser extent TrueSkill. The gains seen by EVMixed are virtually erased with Elo as the base learner, however, an issue seen during evaluation in Section 3.5 as well.

5.3.3 Online classification variants

For EVGen and EVMixed, we investigated three different online classification frameworks - the perceptron [37], Passive-Aggressive algorithms [38], and Confidence-Weighted learning [39] - and evaluated them using a subset of the testbed from section 5.2. Glicko was chosen as the base learner given its exceptional performance in previous evaluations. The following default parameters were used: perceptron ($\alpha = 0.1$), Passive-Aggressive algorithms ($\alpha = 0.1$, $C = 0.001$, and $\eta = 0.9$), and Confidence-Weighted learning ($\alpha = 0.1$, $\eta = 0.9$).

The results are shown in table 5.3. Though performance is largely similar across the different classifiers, the PA-II approach is the most consistent overall (with CW-diag not far behind). Despite its simplicity, the perceptron's performance is nearly equal to that of PA-II, and even edges out PA-II slightly for close games in scrimmages, which

is an interesting observation in and of itself. CW-diag, a much more involved classifier incorporating the notion of uncertainty for each feature's corresponding weight in $\mathbf{w}$, only performs slightly better than the perceptron in tournament games, and worse than it in scrimmages.

5.4 Discussion

In sum, the results show EVMixed consistently outperforming competing approaches, often by great margins, in a variety of experimental settings. In close games especially, EVMixed significantly improves upon the baseline and AllK-EV, underscoring the utility that game-specific data can realize when employed in cases where it is expected to be a meaningful differentiator of skill.

Initially, the subpar performance of EVGen was somewhat surprising given that the only difference between it and EVMixed is the choice of classification mechanism according to a given threshold ϵ. Upon closer examination, the reason for their performance discrepancy becomes clear: the game-specific data used to supplement the feature set $\mathbf{x}$ was *not* weighted according to the strengths of opposing teams in each game, and as such, the game-specific data becomes "noise" when added to the feature set for cases where the games were not otherwise considered close. By selecting our method of classification based on AllK-EV's estimate of each team's skill, taking into account their subgroup-level team chemistry, we can more accurately assess whether or not a particular game is evenly-matched, and if it is, include game-specific features to aid in prediction.

It follows that this is also the reason why both EVGen and EVMixed perform well in close games. Here, because the difference in skill ratings is small, the supplemental feature information tells us something about how two otherwise evenly-matched teams might perform when facing each other. This is also why EVGen and EVMixed have excellent results when little game history has been observed - nearly all games are considered "close" early in the rating process. Once players and teams have competed in enough games such that their ratings cleanly separate the more-skilled from the less-skilled, we almost exclusively echo AllK-EV's predicted label. As such, while there is clearly a benefit to using game-specific features in certain cases, skill ratings still account

for the bulk of the overall prediction task.

Chapter 6

Application of the model to real-world sports

6.1 Lingering questions about TeamSkill performance

Though the suite of TeamSkill approaches described in the previous sections performed well on the MLG Halo 3 dataset, questions about their applicability beyond high-level, competitive Halo, and video games more broadly, remain:

- Is the overall strategy of assessing a team's collective skill as a function of both individual player and group-level cohesion a la TeamSkill viable in real-world, team-based sports? That is, will similar levels of improvement in prediction accuracy be seen when evaluating TeamSkill approaches on real-world, team-based sports, relative to the baseline?
- If there are differences in performance, what factors might account for them? What is it that TeamSkill, or the baseline approaches, fail to capture due to their generalizable framework, if anything?

These and other related questions follow naturally from the TeamSkill approaches and their evaluation using the MLG Halo 3 dataset. This dataset is, in some sense, the *idealized* environment in which one may realize that advantages of incorporating team chemistry in the skill assessment task. Professional Halo players are all exceptionally-skilled individually, but because the kill times (i.e., the number of accurate shots it takes

to kill another player) are longer than most other first-person shooters, those teams that can coordinate their efforts most effectively, such as two teammates creating a numeric advantage by isolating individual players from the opposing team, tend to win. Further, with teams often replacing a player or two after each tournament, which are roughly two to three months apart, the frequency of altered team compositions are sufficient to understand the effects of team chemistry within subgroups of teams without having to account for atrophy of skill among individual players. It is reasonable to assume that these idealized characteristics may not be present across all team-based multi-player video games, let alone the realm of real-world team-based sports.

The nagging questions given above, and subsequent search for an appropriate dataset from real-world team-based sports from which answers might be found, formed the basis for the work of [40] and evaluation detailed in the following section.

6.2 Experiments

Our evaluation of the TeamSkill approaches using data from real-world team-based sports was conducted on a dataset derived from games played during the 2011-2012 National Basketball Association (NBA) regular season [41]. A complete description of the source dataset and the data cleaning and preparation processes are provided in Appendix B. Here, we predicted which team would outscore the other for each "match-up" of players occurring during each game. Match-ups are subsets of games where an instance of a particular configuration of players from each team compete until a "boundary" condition of some kind is reached, such as a player substitution or half-time, at which point the match-up is concluded. Each game may be composed of many such match-ups, resulting in many possible configurations of players for each team, which, in a similar vein as the MLG Halo 3 dataset, help to isolate and understand the effect team chemistry might play with respect to the match-up outcomes.

For EVGen and EVMixed, additional game-specific features such as points scored, rebounds, three-pointers, and fouls were utilized as well (see Table 6.1 for a complete list of game-specific features). The choice of these features corresponds both to the logic outlined in Section 5.3, as well as the work of [42]. After initial evaluation of feature value with respect to match-up prediction accuracy, we quickly found that cumulative

Table 6.1: List of game-specific features used for evaluating the performance of EVGen and EVMixed.

Variable Name
Points per match-up (last d match-up, $k = K$)
Turnovers per match-up (last d match-up, $k = K$)
3 pointers per match-up (last d match-up, $k = K$)
Rebounds per match-up (last d match-up, $k = K$)
Fouls per match-up (last d match-up, $k = K$)
Rate of outscoring opposition (last d match-up, $k = K$)

versions of each feature were useless at best and harmful at worst, an observation which is discussed further in Section 6.3. Additionally, because NBA players remain on the same team throughout the duration of the regular season, versions of features based on individual player histories (i.e., $k = 1$) were not included as they were redundant with respect to $k = K$.

Early analysis of the NBA dataset also revealed large differences in the length of match-ups and, in evaluation, similarly varying levels of accuracy in predicting which team would outscore the other for a given match-up. As such, we decided to alter our previous experimental framework and evaluate each on subsets of the NBA dataset according to minimum match-up length, ignoring those in which neither team scored. The minimum lengths were 30, 60, 90, 120, 150, 200, 250, 300, 350, 400, 450, and 500 seconds. As in previous analyses, the baseline represents the unaltered version of the underlying skill assessment algorithm - Elo, Glicko, or TrueSkill - wherein a team's collective skill is computed as the sum of the skill ratings of each individual player (i.e., TeamSkill-K where $k = 1$). Consequently, all other TeamSkill approaches incorporate some aspect of a team's chemistry as described in sections 3, 4, and 5.

Aside from varying the ϵ and window d parameters for EVGen/EVMixed, all default parameters/settings used for this evaluation are as detailed in previous sections (3.5, 4.5 and 5.3). Due to computational overhead, the ϵ and window parameters for EVGen/EVMixed were limited as the number of match-ups increased (i.e., as the match-up length decreased). For match-ups of 30 to 200 seconds in length, $0.05 \leq \epsilon \leq 0.06$ and $d = 10$ match-ups. For match-ups of 250 seconds, $0.05 \leq \epsilon \leq 0.06$ and $5 \leq d \leq 20$. For match-ups greater than or equal to 300 seconds, $0.02 \leq \epsilon \leq 0.08$ and $5 \leq d \leq 20$.

Table 6.2: Overall accuracy based on maximum observed performance for baseline, team-based approaches, and TeamSkill-AllK-EVMixed.

Min. match-up length	Number of instances	Max. baseline	Max. team-based	Team-based vs. baseline Δ	$\Theta(Z)^*$	Max. EVMixed	EVMixed vs. baseline Δ	$\Theta(Z)^*$
30	19,483	51.98%	52.18%	0.20%	0.350	52.08%	0.10%	0.428
60	14,949	51.80%	52.02%	0.20%	0.356	51.58%	-0.20%	0.653
90	10,601	52.69%	52.63%	-0.10%	0.538	51.38%	-1.30%	0.972
120	7,181	51.92%	52.42%	0.50%	0.274	51.87%	0.00%	0.520
150	5,008	52.74%	53.18%	0.40%	0.330	52.72%	0.00%	0.508
200	2,985	54.34%	55.04%	0.70%	0.292	55.04%	0.70%	0.292
250	1,991	54.50%	55.25%	0.80%	0.316	54.04%	-0.50%	0.613
300	1,383	54.01%	56.04%	2.00%	0.142	55.53%	1.50%	0.211
350	925	53.51%	55.68%	2.20%	0.175	55.14%	1.60%	0.242
400	523	51.05%	55.07%	4.00%	0.097	55.07%	4.00%	0.097
450	252	51.59%	57.54%	6.00%	0.090	57.54%	6.00%	0.090
500	94	54.26%	57.45%	3.20%	0.330	57.45%	3.20%	0.330

*One-tailed Z-test of hypothesis that max. team-based/EVMixed accuracy > max. baseline accuracy

6.2.1 Overall

Table 6.2 provides a high-level overview of the performance of the best-performing baseline approaches relative to that of the best-performing team-based approaches, as well as EVMixed alone given its superior performance in prior evaluations (Section 5.3). Here, we observe that *team-based approaches tend to do at least as well as the baseline for shorter match-ups, and outperform them as the length of the match-up increases*, with EVMixed performing especially well in this regard. Additionally, the margin of this advantage tends to increase as the length of the match-up increases, but it is not found to be statistically significant at the 0.05 level (though approaching it with 0.09 for match-ups of 450 seconds).

Of general note is the observation that the accuracy of all approaches is relatively close to random, and particularly so for the baseline. That well-established skill assessment techniques like Elo, Glicko, and TrueSkill perform as such is a testament to the difficulty of this particular online classification problem.

Table 6.3: Accuracy in "close" games based on maximum observed performance for baseline, team-based approaches, and TeamSkill-AllK-EVMixed.

Min. match-up length	Number of instances	Max. baseline	Max. team-based	Team-based vs. baseline Δ	$\Theta(Z)^*$	Max. EVMixed	EVMixed vs. baseline Δ	$\Theta(Z)^*$
30	3,897	51.25%	51.48%	0.20%	0.420	51.09%	-0.20%	0.554
60	2,990	50.30%	51.41%	1.10%	0.197	49.16%	-1.10%	0.810
90	2,120	51.93%	52.69%	0.80%	0.311	49.91%	-2.00%	0.907
120	1,436	51.39%	53.97%	2.60%	0.083	50.42%	-1.00%	0.699
150	1,002	51.60%	52.99%	1.40%	0.266	52.50%	0.90%	0.344
200	597	53.10%	53.60%	0.50%	0.431	52.09%	-1.00%	0.636
250	398	53.27%	58.29%	5.00%	0.077	53.77%	0.50%	0.443
300	277	50.18%	54.51%	4.30%	0.154	54.51%	4.30%	0.154
350	185	52.43%	52.97%	0.50%	0.458	52.97%	0.50%	0.458
400	105	45.71%	61.91%	16.20%	0.009	61.91%	16.20%	0.009
450	50	44.00%	62.00%	18.00%	0.036	62.00%	18.00%	0.036
500	19	42.11%	73.68%	31.60%	0.024	73.68%	31.60%	0.024

*One-tailed Z-test of hypothesis that max. team-based/EVMixed accuracy > max. baseline accuracy

6.2.2 "Close" match-ups only

Next, we explore the performance of the baseline and team-based approaches for "close" match-ups only, the results of which are given in table 6.3. The same general trend seen in table 6.2 are reflected here as well, albeit with a *much larger margin of advantage for team-based approaches* relative to the baseline than before, *topping out at 31.6%* for EVMixed for match-ups of at least 500 seconds. In several cases, *this advantage is statistically significant* at the 0.05 level (450 and 500 second cases) and, in one case, the 0.01 level (400 seconds).

For match-ups of at least 300 seconds, the EVMixed approach is the source of the greatest performance advantage, further evidence of its viability across different team-based games when teams are closely matched. Also of note is the drop in performance of the baseline approaches as match-up length increases, from 51.25% for 30+ second match-ups to only 42.11% by the time they reach 500 seconds or more. This behavior was also seen in [43] when relatively few games had been observed.

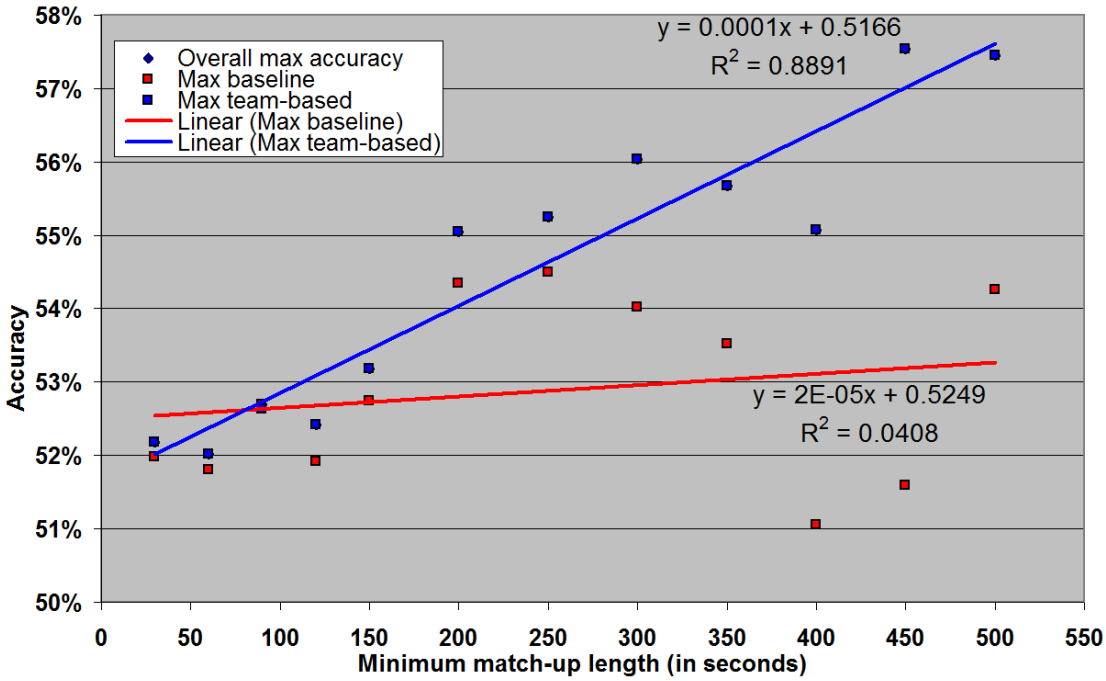


Figure 6.1: Overall maximum team-based/baseline accuracy vs. minimum match-up length.

6.2.3 Relationship between match-up length and predictive performance

Though the margins of advantage for the team-based approaches relative to the baseline in table 6.2 were not found to be sufficiently significant to discard the null hypothesis that the baseline results were at least as good as that of the team-based approaches, it is nonetheless apparent that the *margins tend to grow larger as longer match-ups are evaluated*. In figure 6.1, there is a clear, albeit imperfect, trend ($R^2 = 0.88914$) toward higher overall performance for the team-based approaches as the minimum match-up length increases, while a much weaker trend ($R^2 = 0.04082$) upward exists for the baseline, which is much more variant in its performance.

6.2.4 Performance variance by base learner and TeamSkill approach

For completeness, we include table 6.4, which details the overall performance, and table 6.5, which lists the performance in close games, of each TeamSkill method and the

baseline for each base learner - Elo, Glicko, and TrueSkill - for the complete range of minimum match-up lengths. Because varying the aforementioned ϵ and d parameters were expected to impact the EVGen and EVMixed results, the mean, standard deviation, minimum, and maximum observed performance of EVMixed and the mean performance of EVGen are given as well.

We make several observations. First, these results further underscore the notion that as match-ups become longer, the margin between the best team-based approaches (especially EVMixed) and the baseline grows, with little difference seen between base learner employed. Though certainly not conclusive, *the results suggest a more fundamental connection between minimum match-up length and strong team-based approaches like EVMixed.*

We also note the relationship between the minimum match-up length cutoffs and the points at which each of the TeamSkill-K possibilities for an NBA team of 5 players - TS1, TS2, TS3, TS4, and TS5 in the table notation - achieve their best performance. In the Elo results in Table 6.4, TeamSkill-K subgroup ratings of size $k = 2$ first outperform the baseline at 150 seconds and continue to do so until 450 seconds. The case $k = 3$ doesn't until 300 seconds, then again at 400 and 500 seconds, $k = 4$ does at 300 seconds to 400 seconds, and $k = 5$ does at 350 seconds to 400 seconds. Roughly similar results are seen for Glicko and TrueSkill, that subgroup ratings for cases where $k > 1$ tend to perform better as match-up length increases.

As in Section 3.5, we analyzed the final conservative skill ratings for each base learner. Here, we chose 250 seconds as the minimum match-up length to illustrate the relationship between the choice of base learner and k-sized subgroup ratings. The results show that Glicko and TrueSkill tend to have a wider range of skill ratings relative to Elo, which are concentrated around the mode. Also as in Section 3.5, we note that several subgroups of size $k = 2$ and $k = 3$ greatly exceed any possible summation of their members' individual skill ratings, providing further evidence that subgroup ratings can be useful in helping estimate the overall skill of a team.

TrueSkill stands out as the best overall base learner across almost all minimum match-up lengths for a given TeamSkill variant and choice of base learner, while Glicko doesn't consistently reach parity with the other approaches until minimum match-up length reaches 300 seconds. In 6.5, though, the Elo and TrueSkill learners tend to

perform more and more poorly as longer minimum match-up lengths are seen. In comparison, Glicko is especially consistent, which is reflected in its average performance over all the TeamSkill approaches and test cases from Table 6.4, as well as the low standard deviation of performance for EVMixed given ϵ and d. TrueSkill and Elo's best EVMixed strategies tend to outperform Glicko's, though, albeit with higher performance variation given ϵ and d.

6.3 Discussion

In general, the results demonstrate that the TeamSkill approaches, and EVMixed in particular, can be successfully applied to the task of predicting which team will outscore the other in NBA match-ups in addition to its utility in predicting the outcomes of professional Halo 3 games. *To our knowledge, this is the first such instance in which approaches accounting for group cohesion have been shown to positively impact predictive performance in both team-based video games and real-world team sports.*

As in Chapter 5, EVMixed is the clear standout in our evaluation, particularly for "close" games. The conditional inclusion of game-specific features prove critical during the prediction task when both teams are otherwise evenly-matched, and when they are not, the features are left out to avoid adding "noise" to the feature vector. Of the remaining TeamSkill approaches, AllK also outperforms the baseline in a wide variety of cases, though not to the extent of EVMixed. AllK-EV, OL1, and OL2 perform well in some cases, such as when Glicko is used as the base learner (also as in Chapter 5), but lack the consistency of EVMixed across the entire testbed.

That said, EVMixed's results, and that of the best-performing team-based approaches, were not consistent in and of themselves – the margin of advantage between it and the baseline tended to increase as the minimum length of the match-ups increased, a trend seen repeatedly during the evaluation. It is perhaps most apparent in figure 6.1, with one relatively strong trendline for the best-performing team-based approach and a nearly flat trendline for the baseline. Further, the maximum accuracy seen for any of the non-"close" datasets was 57.5% (EVMixed, TrueSkill base learner, 450+ second match-ups), a far cry from the mid-60% of our previous work with the MLG Halo 3 dataset. What might account for these observations?

The results and conclusions from Chapter 5, as well as the game of basketball itself, suggest several possible explanations. The first is that teamwork in both Halo and the NBA are, in part, functions of the amount of time a particular configuration of players are on the court (or in the game for Halo). That is, *features accounting for group cohesion/team chemistry become more useful to predicting the outcomes of match-ups the longer they play, and the longer they play, the less likely it is that the predicted outcome can be attributed to a sole individual*, the latter of which could reasonably be expected to happen in shorter match-ups.

This *also* may explain why the performance for Halo is higher despite the fact that the same methods are used for the NBA data: *most Halo games are at least 10 minutes long (or 600 seconds), while only 94 match-ups from the entire 2011-2012 NBA season were 500 seconds or more*. Other confounding factors, such as player injuries in the NBA (and their long-term effects), an almost non-existent problem in Halo, may also account for some of the lower performance relative to the Halo results, though it is difficult to compute what precise impact it might have had. Injuries and other hazards of real-world sports are also likely explanations for the observation that game-specific variables derived from the cumulative history of play for a given player or subgroup are less useful than those focused on performance in more recent games, i.e., the window d.

Table 6.4: Accuracy by TeamSkill/baseline approach, base learner, and match-up length (all games).

		Match-up length (in seconds)											
Learner	TeamSkill method	30	60	90	120	150	200	250	300	350	400	450	500
Elo	TS1	51.81%	51.07%	51.23%	51.23%	52.34%	54.07%	54.09%	52.86%	53.30%	49.90%	51.59%	46.81%
	TS2	51.70%	51.01%	50.60%	50.97%	52.38%	54.07%	54.80%	53.65%	54.49%	50.86%	50.40%	47.87%
	TS3	51.54%	50.69%	50.35%	50.84%	51.98%	53.63%	53.39%	53.58%	52.86%	50.48%	51.59%	48.94%
	TS4	51.13%	50.65%	50.53%	50.91%	51.84%	53.53%	52.99%	53.07%	53.95%	52.20%	50.00%	46.81%
	TS5	50.88%	50.63%	50.51%	50.69%	51.90%	53.17%	51.58%	51.63%	54.27%	50.29%	48.41%	46.81%
	AllK	51.62%	51.13%	51.08%	51.55%	52.66%	53.90%	54.70%	54.01%	53.62%	49.90%	53.57%	50.00%
	AllKEV	51.66%	50.94%	50.71%	51.05%	52.44%	54.17%	53.79%	53.07%	53.19%	50.10%	51.19%	45.74%
	AllKLS	51.39%	50.82%	50.44%	49.88%	51.98%	53.90%	53.09%	53.22%	52.76%	51.24%	51.59%	47.87%
	OL1	51.83%	51.05%	51.18%	51.13%	52.28%	54.10%	54.09%	52.93%	53.30%	49.90%	51.19%	46.81%
	OL2	51.80%	51.05%	51.12%	50.90%	52.26%	54.41%	54.34%	52.49%	53.08%	50.10%	51.59%	45.74%
	OL3	51.63%	50.95%	50.71%	50.94%	52.46%	54.24%	53.79%	53.15%	53.08%	50.10%	51.59%	45.74%
	EVGen (μ)	49.74%	49.34%	49.17%	50.70%	50.34%	50.32%	50.21%	50.98%	50.05%	49.81%	47.92%	55.59%
	EVMixed (μ)	51.07%	50.33%	50.17%	51.70%	52.36%	54.68%	52.75%	53.96%	53.74%	52.05%	52.37%	53.87%
	EVMixed (σ^2)	0.14%	0.06%	0.15%	0.16%	0.10%	0.18%	0.64%	0.73%	0.54%	1.29%	2.18%	1.93%
	EVMixed (min)	50.92%	50.28%	50.00%	51.57%	52.28%	54.51%	51.78%	52.71%	52.65%	49.52%	48.41%	51.06%
	EVMixed (max)	51.20%	50.40%	50.29%	51.87%	52.48%	54.87%	53.74%	55.53%	54.81%	55.07%	55.95%	57.45%
Glicko	TS1	51.85%	51.56%	52.00%	51.59%	52.18%	53.84%	53.44%	54.01%	53.51%	49.52%	48.81%	54.26%
	TS2	50.54%	50.28%	50.49%	50.52%	50.62%	51.76%	50.98%	52.35%	53.73%	52.39%	46.03%	51.06%
	TS3	50.53%	50.58%	49.99%	49.38%	50.84%	50.32%	51.68%	52.13%	52.76%	52.77%	46.03%	50.00%
	TS4	50.20%	49.78%	50.17%	50.17%	49.74%	52.16%	52.13%	52.13%	53.41%	51.05%	50.40%	48.94%
	TS5	51.54%	51.33%	51.15%	51.61%	51.46%	53.33%	52.99%	52.86%	52.86%	50.48%	45.63%	44.68%
	AllK	51.58%	52.02%	52.10%	52.42%	52.34%	54.17%	55.25%	56.04%	55.68%	51.43%	50.00%	50.00%
	AllKEV	50.71%	50.63%	50.21%	50.70%	51.32%	51.86%	51.73%	53.15%	53.62%	53.35%	49.21%	51.06%
	AllKLS	49.77%	49.72%	49.72%	49.14%	48.20%	48.84%	48.82%	47.29%	45.51%	47.99%	48.81%	53.19%
	OL1	51.73%	51.37%	51.77%	51.68%	51.78%	53.90%	53.94%	53.15%	53.08%	51.82%	48.41%	54.26%
	OL2	51.67%	51.07%	51.50%	51.32%	51.94%	53.07%	52.39%	53.15%	53.73%	53.54%	49.21%	51.06%
	OL3	50.75%	50.67%	50.03%	50.61%	51.58%	51.86%	51.68%	53.22%	53.41%	53.35%	49.21%	51.06%
	EVGen (μ)	49.58%	49.56%	49.80%	50.94%	49.32%	50.15%	50.73%	50.58%	49.57%	49.76%	49.90%	54.52%
	EVMixed (μ)	50.76%	50.65%	50.29%	50.77%	51.48%	52.15%	52.50%	54.14%	54.44%	54.63%	53.04%	56.79%
	EVMixed (σ^2)	0.01%	0.01%	0.01%	0.02%	0.03%	0.02%	0.19%	0.16%	0.14%	0.17%	0.39%	0.52%
	EVMixed (min)	50.75%	50.65%	50.28%	50.76%	51.46%	52.13%	52.18%	53.87%	54.05%	54.11%	52.38%	56.38%
	EVMixed (max)	50.76%	50.67%	50.31%	50.80%	51.52%	52.16%	52.74%	54.45%	54.59%	54.88%	53.57%	57.45%
TrueSkill	TS1	51.98%	51.80%	52.69%	51.91%	52.74%	54.34%	54.50%	53.65%	53.51%	51.05%	50.40%	46.81%
	TS2	51.75%	51.45%	51.26%	51.26%	52.76%	53.97%	54.85%	54.16%	54.81%	50.67%	50.40%	47.87%
	TS3	51.72%	50.88%	50.24%	50.81%	52.22%	53.90%	53.19%	53.00%	52.76%	50.86%	52.78%	52.13%
	TS4	51.43%	50.94%	50.84%	50.54%	51.82%	53.67%	52.89%	53.58%	54.49%	52.39%	50.40%	50.00%
	TS5	51.72%	51.46%	51.28%	51.66%	52.22%	53.00%	52.79%	51.70%	52.86%	51.05%	49.60%	46.81%
	AllK	51.87%	51.67%	51.81%	51.97%	53.17%	54.17%	54.90%	54.52%	54.49%	50.86%	52.78%	51.06%
	AllKEV	52.15%	51.59%	51.34%	51.57%	52.74%	54.47%	54.14%	54.59%	53.84%	51.63%	50.40%	47.87%
	AllKLS	52.00%	50.94%	51.05%	51.08%	52.44%	54.41%	53.29%	55.10%	54.27%	50.48%	51.19%	50.00%
	OL1	51.94%	51.77%	52.63%	51.87%	52.68%	54.44%	54.50%	53.94%	53.41%	50.67%	51.19%	46.81%
	OL2	52.18%	51.89%	52.35%	52.30%	52.50%	54.57%	54.39%	54.23%	53.41%	51.82%	50.00%	47.87%
	OL3	52.13%	51.54%	51.48%	51.61%	52.72%	54.54%	54.29%	54.37%	53.73%	51.63%	50.00%	46.81%
	EVGen (μ)	49.94%	49.82%	49.49%	50.83%	49.72%	50.59%	50.84%	50.78%	49.41%	49.71%	48.21%	54.79%
	EVMixed (μ)	52.04%	51.52%	51.33%	51.72%	52.64%	54.96%	53.60%	54.46%	53.91%	52.82%	54.69%	53.07%
	EVMixed (σ^2)	0.04%	0.08%	0.04%	0.16%	0.06%	0.07%	0.17%	0.35%	0.76%	0.85%	1.45%	2.16%
	EVMixed (min)	52.00%	51.43%	51.30%	51.55%	52.60%	54.91%	53.39%	53.87%	52.11%	51.24%	52.78%	48.94%
	EVMixed (max)	52.08%	51.58%	51.38%	51.87%	52.72%	55.04%	54.04%	55.17%	55.14%	54.30%	57.54%	57.45%

Table 6.5: Accuracy by TeamSkill/baseline approach, base learner, and match-up length (close games).

		Match-up length (in seconds)											
Learner	TeamSkill method	30	60	90	120	150	200	250	300	350	400	450	500
Elo	TS1	51.22%	50.30%	50.80%	49.93%	49.60%	49.75%	52.26%	46.21%	45.95%	34.29%	44.00%	26.32%
	TS2	49.96%	49.97%	48.54%	48.47%	50.60%	49.25%	55.28%	48.38%	47.57%	40.95%	36.00%	26.32%
	TS3	49.68%	49.00%	48.25%	51.04%	49.40%	47.91%	48.74%	46.57%	40.00%	41.90%	40.00%	31.58%
	TS4	51.09%	48.76%	47.97%	47.84%	49.80%	47.40%	51.01%	45.49%	46.49%	44.76%	34.00%	31.58%
	TS5	49.47%	50.50%	48.49%	49.93%	50.10%	49.25%	48.24%	46.93%	45.41%	39.05%	32.00%	31.58%
	AllK	50.42%	49.70%	49.53%	50.21%	50.10%	48.41%	53.52%	49.46%	42.16%	36.19%	44.00%	42.11%
	AllKEV	49.94%	49.40%	48.73%	48.47%	50.30%	49.08%	50.25%	46.57%	43.24%	37.14%	40.00%	15.79%
	AllKLS	50.65%	50.37%	49.95%	47.77%	51.90%	47.40%	53.02%	48.74%	45.95%	42.86%	44.00%	26.32%
	OL1	51.27%	49.36%	49.67%	48.75%	49.20%	48.74%	51.26%	46.21%	44.86%	36.19%	40.00%	21.05%
	OL2	51.12%	49.30%	49.48%	47.63%	49.40%	50.08%	52.76%	44.04%	44.32%	37.14%	42.00%	15.79%
	OL3	49.86%	49.40%	48.58%	48.12%	50.50%	49.41%	50.25%	46.93%	43.24%	37.14%	42.00%	15.79%
	EVGen (μ)	48.94%	48.73%	50.05%	49.44%	50.30%	52.09%	49.18%	49.73%	51.08%	57.14%	59.00%	64.47%
	EVMixed (μ)	48.95%	48.66%	49.83%	49.37%	50.53%	52.09%	49.18%	49.99%	51.24%	57.33%	59.00%	64.47%
	EVMixed (σ^2)	0.15%	0.09%	0.10%	0.12%	0.12%	0.00%	0.39%	0.88%	0.75%	2.64%	2.26%	2.30%
	EVMixed (min)	48.78%	48.56%	49.72%	49.23%	50.40%	52.09%	48.74%	48.74%	49.73%	54.29%	56.00%	63.16%
	EVMixed (max)	49.06%	48.73%	49.91%	49.44%	50.60%	52.09%	49.75%	51.62%	52.97%	61.90%	62.00%	68.42%
Glicko	TS1	51.24%	50.23%	51.37%	49.72%	51.60%	53.10%	52.51%	50.18%	43.78%	45.71%	40.00%	42.11%
	TS2	49.40%	48.83%	47.74%	49.44%	49.30%	47.74%	47.74%	44.04%	42.70%	43.81%	36.00%	42.11%
	TS3	49.91%	49.26%	48.73%	48.47%	51.40%	44.22%	49.25%	44.04%	45.41%	41.90%	32.00%	52.63%
	TS4	49.06%	47.09%	49.06%	47.98%	48.70%	49.08%	49.50%	45.85%	45.41%	40.95%	38.00%	36.84%
	TS5	49.35%	50.17%	50.38%	49.72%	48.90%	49.25%	51.01%	48.74%	45.95%	44.76%	24.00%	31.58%
	AllK	50.60%	51.40%	52.55%	53.97%	52.99%	51.26%	58.29%	50.90%	49.19%	41.90%	32.00%	42.11%
	AllKEV	49.19%	49.00%	47.41%	49.65%	51.90%	45.73%	49.75%	42.96%	46.49%	45.71%	38.00%	42.11%
	AllKLS	48.94%	47.89%	49.10%	47.42%	48.60%	45.39%	52.01%	43.68%	42.16%	43.81%	48.00%	47.37%
	OL1	49.29%	48.56%	49.01%	50.77%	50.00%	49.92%	54.27%	47.65%	41.08%	43.81%	34.00%	42.11%
	OL2	49.27%	47.83%	48.96%	49.79%	50.40%	47.24%	51.01%	43.68%	47.57%	46.67%	38.00%	42.11%
	OL3	49.42%	48.93%	46.93%	49.44%	52.50%	46.06%	50.25%	43.32%	46.49%	46.67%	38.00%	42.11%
	EVGen (μ)	50.99%	47.36%	50.80%	51.32%	49.50%	53.60%	51.57%	51.81%	49.73%	49.29%	48.50%	59.21%
	EVMixed (μ)	49.41%	49.11%	47.89%	50.42%	52.43%	47.68%	53.18%	47.23%	50.90%	52.62%	53.69%	68.42%
	EVMixed (σ^2)	0.04%	0.05%	0.03%	0.00%	0.06%	0.10%	0.43%	0.43%	0.41%	0.42%	0.73%	0.00%
	EVMixed (min)	49.37%	49.06%	47.88%	50.42%	52.40%	47.57%	52.51%	46.57%	50.27%	52.38%	52.00%	68.42%
	EVMixed (max)	49.45%	49.16%	47.92%	50.42%	52.50%	47.74%	53.77%	48.38%	51.35%	53.33%	54.00%	68.42%
TrueSkill	TS1	49.60%	49.80%	51.93%	51.39%	49.70%	50.75%	53.27%	49.82%	52.43%	44.76%	34.00%	31.58%
	TS2	51.14%	48.70%	49.34%	47.63%	50.60%	49.75%	55.03%	49.46%	51.89%	44.76%	40.00%	21.05%
	TS3	50.91%	48.23%	47.78%	46.80%	50.00%	49.92%	49.75%	45.49%	45.41%	45.71%	46.00%	31.58%
	TS4	50.14%	48.19%	47.74%	45.61%	51.60%	48.91%	53.52%	46.21%	47.03%	49.52%	38.00%	26.32%
	TS5	50.58%	50.13%	49.62%	51.18%	49.30%	49.58%	48.74%	44.77%	43.78%	40.00%	40.00%	21.05%
	AllK	50.37%	49.30%	50.24%	48.12%	49.40%	47.91%	56.03%	50.18%	48.65%	43.81%	38.00%	31.58%
	AllKEV	51.32%	48.70%	48.87%	46.94%	49.80%	51.09%	52.26%	51.99%	46.49%	46.67%	40.00%	15.79%
	AllKLS	51.48%	49.13%	49.34%	47.63%	50.20%	51.26%	49.75%	52.71%	49.73%	40.95%	42.00%	26.32%
	OL1	49.19%	48.86%	52.69%	47.77%	50.40%	49.75%	53.02%	51.26%	45.95%	45.71%	42.00%	15.79%
	OL2	49.73%	49.97%	51.37%	48.68%	50.00%	50.59%	52.51%	50.18%	44.86%	45.71%	38.00%	15.79%
	OL3	51.32%	48.63%	49.29%	47.28%	50.10%	51.42%	52.76%	50.90%	45.95%	46.67%	38.00%	10.53%
	EVGen (μ)	50.04%	49.10%	48.87%	47.63%	52.00%	49.92%	50.31%	51.35%	46.35%	54.76%	57.00%	69.74%
	EVMixed (μ)	51.06%	48.76%	48.81%	47.73%	52.26%	50.25%	50.65%	51.76%	48.36%	53.99%	58.88%	68.93%
	EVMixed (σ^2)	0.03%	0.03%	0.07%	0.41%	0.21%	0.17%	1.37%	1.47%	1.63%	2.01%	2.15%	3.65%
	EVMixed (min)	51.04%	48.73%	48.73%	47.42%	52.10%	50.08%	48.99%	48.74%	45.41%	49.52%	54.00%	57.89%
	EVMixed (max)	51.09%	48.80%	48.87%	48.19%	52.50%	50.42%	53.52%	54.51%	51.35%	57.14%	62.00%	73.68%

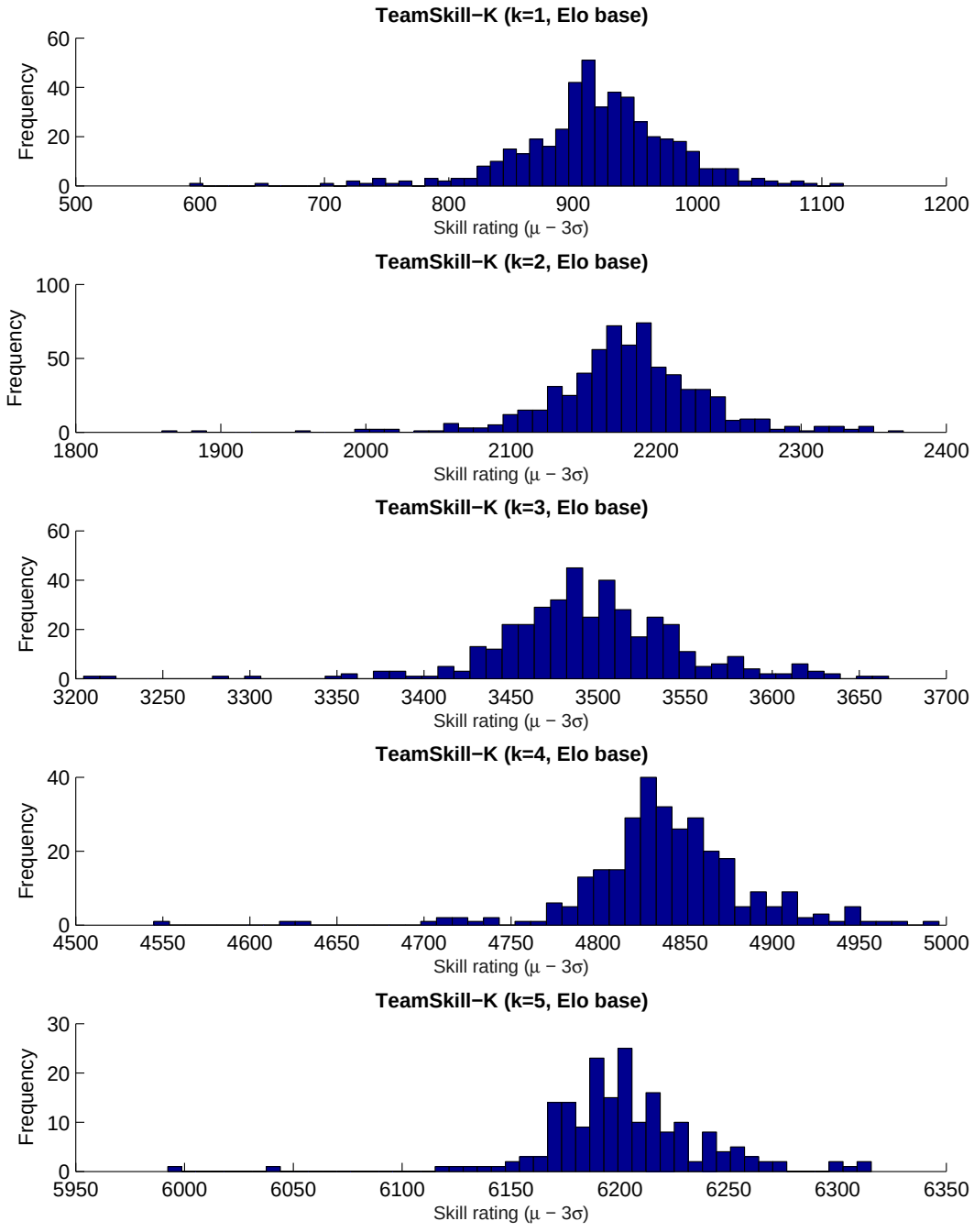


Figure 6.2: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with Elo ($\beta = 193.4364$, $\mu_0 = 1500k$, $\sigma_0^2 = \beta^2 k$) as the base learner, minimum match-up length of 250 seconds.

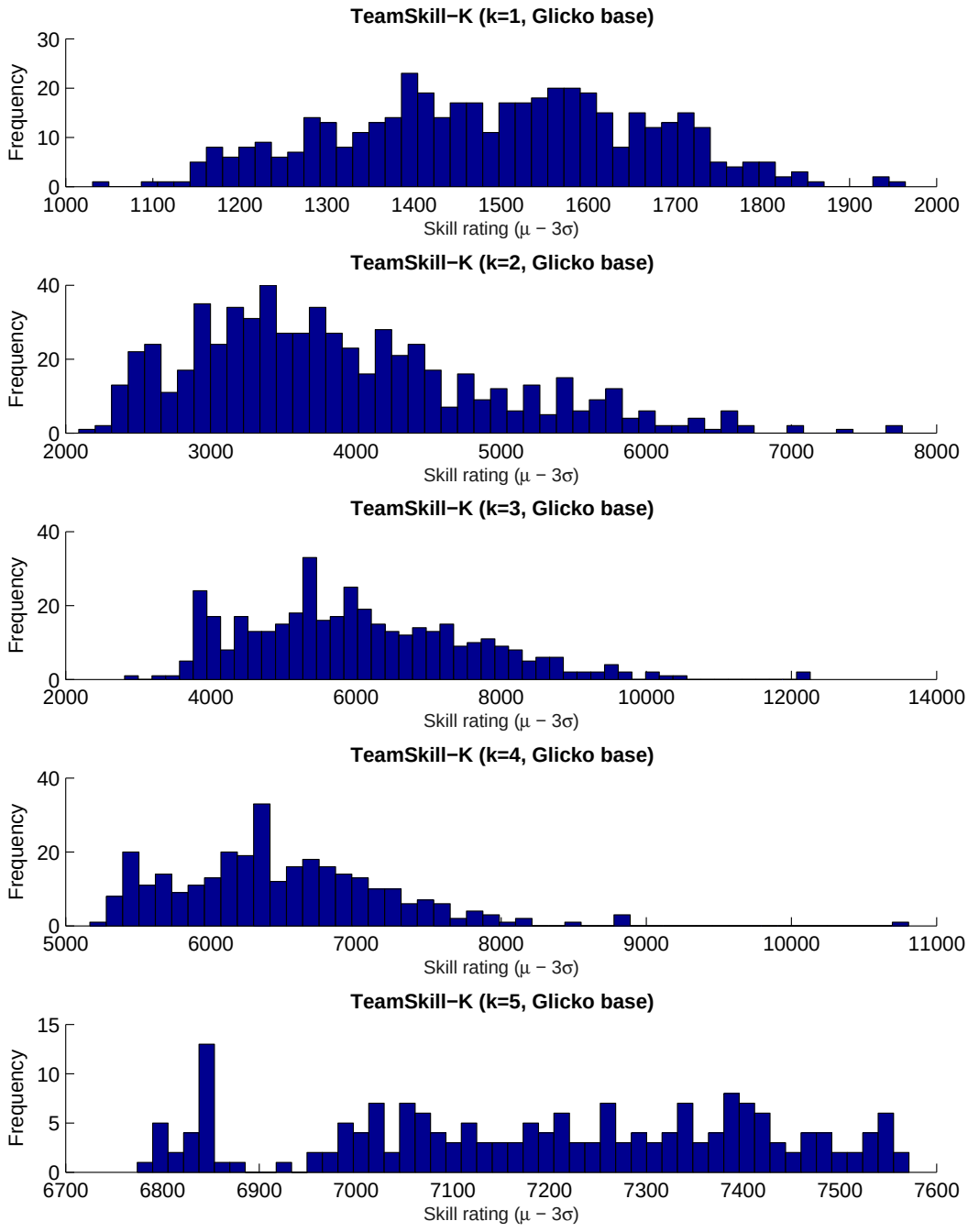


Figure 6.3: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with Glicko ($\mu_0 = 1500k, \sigma_0^2 = 100^2 k$) as the base learner, minimum match-up length of 250 seconds.

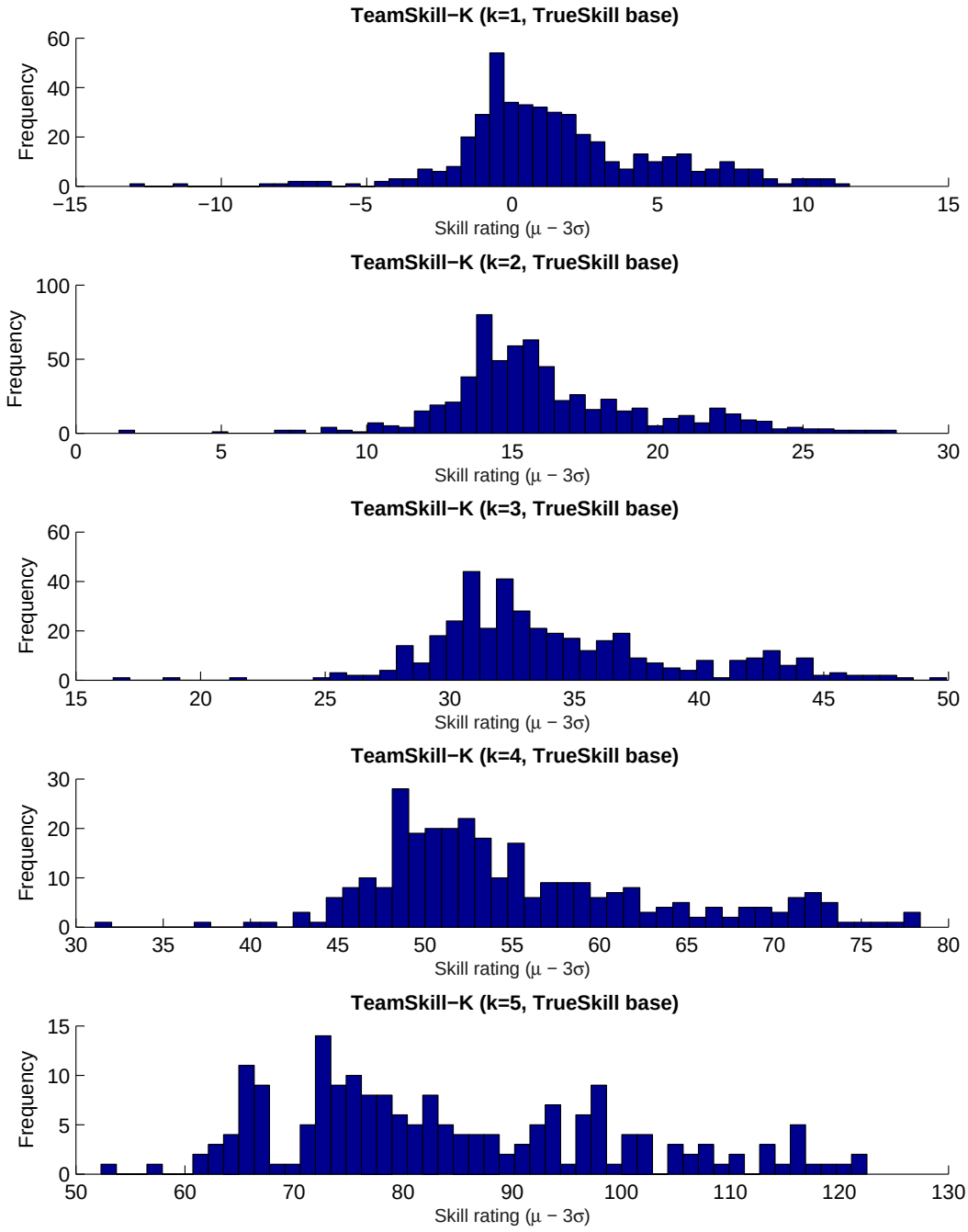


Figure 6.4: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with TrueSkill ($\mu_0 = 25k$, $\sigma_0^2 = (\mu_0/3)^2 k$) as the base learner, minimum match-up length of 250 seconds.

Chapter 7

Conclusion and future directions

7.1 Summary

This thesis presents a wide-ranging analysis of several techniques which account for team chemistry during the skill assessment process. In online multi-player team-based games especially, such approaches are necessary in order to more accurately assess the relative strengths of teams when competitions are dynamically generated, a near constant task in automated matchmaking. Failing to do so risks matching players and teams up with one another whose shared prior game history isn't factored into their pairing, resulting in unfair games where one team is significantly more skilled than the other in ways their individual skill alone wouldn't suggest. The history of research in sports and organizational psychology are aligned in the belief that team chemistry is an important component of a team's collective performance - our work leverages this viewpoint and demonstrates how approaches built to account for such information can improve over existing techniques.

The problem of modeling team chemistry based on the limited set of available information can be addressed by reframing the entity ranked as a subgroup of players, rather than individuals alone. This model treats team chemistry as a latent factor comingled with skill, and as such, the performances of subgroups provide clues as to the cohesion of a team overall and their respective skill ratings combined in some way in order to estimate the skill of a team. In doing so, a new problem is created: how to best aggregate a dynamic set of subgroup skill ratings in order to improve prediction

accuracy. Using this framework, "base" learners can be constructed using existing skill assessment techniques like Elo, Glicko, and TrueSkill, and used in ensemble. This strategy retains both the online setting and broad applicability of the baseline approaches while accounting for team chemistry during the skill assessment process.

A set of naïve aggregation approaches are proposed. Two of them, TeamSkill-AllK and TeamSkill-AllK-EV, consistently outperform the baseline using data from games between professional Halo 3 players. AllK and AllK-EV share the idea that all available subgroup ratings should be used when estimating the skills of teams (including those of individual players), rather than logical subsets such as summing individual player ratings or only using the largest subgroups with prior game history as the measure of a team's skill. AllK-EV is found to perform especially well in "close" games, i.e., games where the likelihood of either team winning is near 50%, a finding consistent with the notion that games between otherwise evenly-matched teams may be decided based on team chemistry.

Several methods for managing the aggregation weights for the ratings corresponding to each set of subgroups of size k are described. Collectively, their goal is to improve on the more naïve aggregation methods and address the dynamic feature space and subgroup history problems by tuning their aggregation weights based on their TeamSkill-K performance on unseen game data. From evaluation, it is demonstrated that the best of these approaches fare no better than their best-performing naïve counterparts, namely TeamSkill-AllK-EV. The primary drawback of the proposed online weight management approaches is the discounting of weights for larger-k subgroup ratings when little game history has been observed, and slowness in attributing more importance when more history is available, a problem inadvertently addressed by the best naïve aggregation approaches, whose notion of subgroup importance is constant across all possible values for k.

A mechanism for including game-specific features is proposed. Here, TeamSkill-AllK-EV's predicted label of the winning team is added to a feature vector where the other features are a selected set of game-specific metrics, which is then used in a standard online classification framework to make the final prediction as to the outcome of

the game. Regardless of the type of game under consideration, any desired set of game-specific metrics may be used in this framework, thus retaining much of the generalizability of previous skill assessment approaches while simultaneously building in the capacity to make use of "in-game" data to further improve prediction accuracy *and* accounting for team chemistry. Two variants of this general strategy, TeamSkill-AllK-EVGen and TeamSkill-AllK-EVMixed, are introduced. While EVGen employs the entire feature vector to predict the outcome of every unseen game, EVMixed only elects to use the game-specific features if the teams competing are considered relatively evenly-matched (within some threshold ϵ), reverting to the predicted label of TeamSkill-AllK-EV alone if they aren't. This more nuanced approach consistently outperforms all previous TeamSkill approaches and baseline methods, often by significant margins, and especially so in close games. Because of this, we conclude that in addition to team chemistry's utility in "deciding" close games, game-specific features are only useful when teams are relatively evenly-matched, or more generally, that for any feature to be of value in this context, it must account for the skills of the teams based on their prior opponents' teams' skill, otherwise their inclusion adds "noise" to the classification problem.

EVMixed, and EVGen to a lesser extent, are also found to help address the "cold start" problem in skill assessment - the task of predicting the outcomes of games when little history has been observed. In such situations, the skill ratings for teams are close to each other, and therefore game-specific features are an improved, albeit imperfect, estimate of team skill until more history is available. As such, rather than the gradual improvement seen over time by the previous TeamSkill and baseline approaches, EVGen and EVMixed perform exceptionally well from the outset, with EVMixed holding this advantage for the duration of the evaluation.

In an effort to assess the applicability of TeamSkill beyond online multi-player team-based video games, an evaluation of the entire suite of TeamSkill approaches is conducted on a dataset containing and entire season of games played in the National Basketball Association (NBA) from 2011-2012. Again, EVMixed consistently outperforms the other TeamSkill approaches and the baseline in nearly all test cases, the first known case in which skill assessment techniques accounting for team chemistry have been shown to transfer successfully from team-based video games to real-world team sports. The results also provide further validation of the idea that team chemistry can be a valuable factor

in predicting the outcomes of games, whether in video games or in real-world sports, as well as the general observation that these factors exist at all in two seemingly very different competitive contexts.

It is also observed that, in general, the performance of the TeamSkill and baseline approaches is much lower than their corresponding performance on Halo 3 dataset, which is attributed to the decreased length of time in which a set of players competes prior to a substitution or other boundary condition in NBA games versus Halo 3 games. The conclusion is that the longer their time competing together, the more valued latent factors like team chemistry become with respect to predicting the outcomes of games. This conclusion is further underscored by the observation that the performance of EVMixed diverges the least from its Halo 3 results when considering the history of the longest NBA match-ups and that the shortest Halo 3 games tend to be longer than the longest NBA match-ups. The correlation between match-up length and maximum observed performance for team-based approaches, particularly EVMixed, is found to be very strong ($R^2 = 0.88914$).

Taken together, the TeamSkill suite provides strong evidence that skill assessment in team-based games can be improved by accounting for team chemistry, and that game-specific features can also be included to further improve prediction accuracy without sacrificing the hard constraints of the general case. Among the entire suite of proposed approaches, EVMixed emerges as the best performing TeamSkill variant by a statistically-significant margin, and as such is the strongest TeamSkill candidate for broader implementation.

7.2 Future work

As with all research, additional questions and corresponding areas of potential future work were generated from the TeamSkill research effort. While this the work of this thesis covers much territory with respect to the problem of skill assessment for team-based games, and proposes a number of solutions to key problems in this space, there still remain many untackled questions and unaddressed issues.

The results from Chapter 3 highlight the crucial role played by skill variance in

estimating the skill of a group of players. Here, Elo's reliance on a constant skill variance parameter exacerbates pre-existing issues with a "sum of parts" framework due to the increase in subgroups under consideration, each of which may have very different amounts of available game history. The end result is that Elo does not appear to benefit at all from the additional team chemistry information afforded by the subgroup-orientated skill rating strategy. Methods for more fully addressing such an issue could consist of maintaining a prior distribution over the skill variance itself or using a mixture model for estimating variance of skill. Extensions to Glicko or TrueSkill of this kind could result in techniques that can better assimilate new observations with prior beliefs in order to generate superior predictions.

Intuitively, because this variance is a function of that subgroup's prior game history, a dynamic skill variance parameter is meant to account for the differing levels of confidence we might have for any individual subgroup skill rating. In practice, however, this alone is insufficient to address the issues that arise when trying to aggregate a potentially large number of subgroup ratings in the presence of the subgroup history problem. The work of Chapter 4 was intended to ameliorate this issue, but extensive evaluation demonstrated the inability of logical dynamic weight management methods, motivated by related strategies in traditional ensemble learning spaces, to improve on naïve approaches. A different strategy may be required, with potential options including assigning a distribution to each aggregation weight (similar to Confidence-Weighted learning), or revising the loss function over the aggregation weights to assign less value to subgroup rating sets with less "evidence" (i.e., when the set has fewer members).

Although Chapter 5 clearly demonstrates the utility of game-specific data in predicting the outcomes of games, more could be done to enhance the EVMixed model itself, perhaps by introducing a mechanism by which the close-game threshold parameter ϵ can vary over time. It may also be interesting to explore whether or not ϵ is needed at all, and instead "normalize" the game-specific metrics for each team based on their respective TeamSkill-AllK-EV skill ratings as a means of "correcting" each metric with respect to the strengths of prior opponents which produced it.

Additional future work could also include evaluating the TeamSkill suite on additional datasets for team-based games, both from other online multi-player video games, such as League of Legends or Defense of the Ancients 2, or real-world sports like hockey,

soccer, or rugby. Another direction is to predict the scores of games or match-ups themselves instead of a simple boolean "team i is predicted to win" approach. Doing so would make TeamSkill viable for predicting the outcomes of entire NBA games, for example, as well as other games where different subsets of teams compete against each other throughout the duration of games.

This would also aid in the problem of optimal team formation, another potential area of future work. By identifying subgroups of players expected to play especially well together due to their shared team chemistry and outscore their opposition consistently, coaches and managers might move toward recruiting groups of players in an effort to maximize their likelihood of winning a certain number of games throughout the season. A sufficiently nuanced version of this extension might also be able to help coaches decide who should replace whom during a game so as to give their team the best chance at outscoring their opponents in individual match-ups.

In summary, we invite other researchers to more fully explore the team chemistry-minded skill assessment foundation detailed in this thesis. Team-based games are everywhere you look - in sports, in video games, in businesses, in the military - and so efforts to improve techniques for estimating the skills of teams are of critical importance.

References

[1] R. Herbrich and T. Graepel. Trueskill: A bayesian skill rating system. Technical Report MSR-TR-2006-80, Microsoft Research, 2006.

[2] D. G. Ancona and D. Nadler. Teamwork at the top: creating high performing executive teams. Working papers 3029-89, Massachusetts Institute of Technology (MIT), Sloan School of Management, 1989.

[3] D. G. Mathiasen. Groups that work (and those that don't): Creating conditions for effective teamwork. *Journal of Policy Analysis and Management*, 10(4):711–712, 1991.

[4] R. S. Wellins, W. C. Byham, and G. R. Dixon. *Inside teams: How 20 world-class organizations are winning through teamwork.* Jossey-Bass, 1994.

[5] W. J. Mayo, C. H. Mayo, and W. W. Mayo. Concept of group practice of medicine, 1892.

[6] L. P. Casalino, K. J. Devers, T. K. Lake, M. Reed, and J. J. Stoddard. Benefits of and barriers to large medical group practice in the united states. *Archives of Internal Medicine*, 163(16):1958–1964, 2003.

[7] M. A. West and B. C. Poulton. A failure of function: teamwork in primary health care. *Journal of Interprofessional Care*, 11(2):205–216, 1997.

[8] D. Heide, E. Auf, and R. L. Irwin. *Disaster response: principles of preparation and coordination.* Mosby, 1989.

[9] M. Lewis. *Moneyball: The art of winning an unfair game.* WW Norton and Company, 2004.

[10] P. Campos and J. Chait. Sabermetrics for football. `http://www.nytimes.com/2004/12/12/magazine/12SABER.html?_r=0`, Dec 2004. The New York Times.

[11] C. Pronman. Hockey prospectus. `http://www.puckprospectus.com/`, Aug 2013. Prospectus Entertainment Ventures.

[12] C. C. Mcclintock. Pair of students get jobs in pros. `http://www.thecrimson.com/article/2010/12/3/hsacstudents-frontoffice-120310/`, Dec 2010. The Harvard Crimson.

[13] L.L. Thurstone. Psychophysical analysis. *American Journal of Psychology*, 38:368–389, 1927.

[14] R. A. Bradley and M. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[15] H. A. David. The method of paired comparisons. In *Proceedings of the Fifth Conference on the Design of Experiments in Army Research, Development, and Testing*, pages 1–16, Fort Detrick, Frederick, Maryland, 1960.

[16] A. Elo. *The Rating of Chess Players, Past and Present.* Arco Publishing, New York, 1978.

[17] K. Harkness. *The Official Blue Book and Encyclopedia of Chess.* D. McKay Company, 1957. Published for the U.S. Chess Federation.

[18] M. Glickman. *Paired Comparison Model with Time-Varying Parameters. PhD thesis.* Harvard University, Cambridge, Massachusetts, 1993.

[19] M. Glickman. Parameter estimation in large dynamic paired comparison experiments. *Applied Statistics*, 48:377–394, 1999.

[20] R. Coulom. Whole-history rating: A bayesian rating system for players of time-varying strength. In *Computer and Games*, pages 113–124, Beijing, China, 2008.

[21] T. Huang, C. Lin, and R. Weng. Ranking individuals by group comparisons. *Journal of Machine Learning Research*, 9:2187–2216, 2008.

[22] J. Langford A. Beygelzimer and P. Ravikumar. Error-correcting tournaments. In *Algorithmic Learning Theory*, volume 5809 of *Lecture Notes in Computer Science*, pages 247–262. Springer, 2009.

[23] A. Birlutiu and T. Heskes. Expectation propagation for rating players in sports competitions. In *Proceedings of the 11th European Symposium on Principles of Knowledge Discovery in Databases (PKDD-07)*, pages 374–381, Warsaw, Poland, 2007. Springer-Verlag.

[24] J. E. Menke, C. S. Reese, and T. R. Martinez. Hierarchical models for estimating individual ratings from group competitions. *American Statistical Association (in preparation)*, 2007.

[25] J. E. Menke and T. R. Martinez. A bradley-terry artificial neural network model for individual ratings in group competitions. *Neural Computing and Applications*, 17:175–186, 2008.

[26] L. S. Shapley. A value for n-person games. *Contributions to the Theory of Games (Annals of Mathematics Studies)*, 2(28):307–317, 1953.

[27] D. Yukelson. Principles of effective team building interventions in sport: A direct services approach at penn state university. *Journal of Applied Sport Psychology*, 9(1):73–96, 1997.

[28] R. Martens. *Coaches guide to sport psychology.* Human Kinetics, Champaign, Illinois, 1987.

[29] D. R. Forsyth and J. L. Burnett. Group processes. In E. R. Baumeister and E. Finkel, editors, *Advanced social psychology*, pages 495–534. New York: Cambridge, 2010.

[30] Aristotle and W. D. Ross. Book viii: Eta. In *Metaphysics: a revised text with introduction and commentary.* Clarendon Press, 1924.

[31] Aristotle and H. Tredennick. *Aristotle in twenty-three volumes: The Metaphysics : books I-IX*, volume 17 of *Aristotle*. Harvard University Press, 1980.

[32] L. Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1–39, 2010.

[33] A. Zander. *Making groups effective*. Jossey-Bass, San Francisco, California, 1982.

[34] Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Data Bases*, volume 1215, pages 487–499, 1994.

[35] C. DeLong, N. Pathak, K. Erickson, E. Perrino, K. Shim, and J. Srivastava. Teamskill: Modeling team chemistry in online multi-player games. In *Advances in Knowledge Discovery and Data Mining*, pages 519–531, Shenzhen, China, 2011. Pacific-Asia Conference on Knowledge Discovery and Data Mining.

[36] C. DeLong, K. Erickson, E. Perrino, K. J. Shim, and N. Pathak. Project halofit data repository. `http://halofit.org/resources.php#datasets`, 2009. University of Minnesota, Department of Computer Science, Minneapolis, MN, USA.

[37] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[38] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551585, 2006.

[39] K. Crammer, M. Dredze, and F. Pereira. Exact convex confidence-weighted learning. *Advances in Neural Information Processing Systems*, 21:345–352, 2009.

[40] C. DeLong, L. Terveen, and J. Srivastava. Teamskill and the nba: Applying lessons from virtual worlds to the real-world. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, Niagara Falls, Canada, 2013.

[41] A. Barzilai. Basketballvalue.com. `http://basketballvalue.com/index.php`, 2012.

[42] D. T. Rosenbaum. Measuring how nba players help their teams win. http://www.82games.com/comm30.htm, Apr 2004. 82games.com.

[43] C. DeLong and J. Srivastava. Teamskill evolved: Mixed classification schemes for team-based multi-player games. In *Advances in Knowledge Discovery and Data Mining*, pages 26–37, Kuala Lumpur, Malaysia, 2012. Pacific-Asia Conference on Knowledge Discovery and Data Mining.

[44] S. DiGiovanni and M. Sepso. Major league gaming. http://www.mlgpro.com, 2013.

[45] Bungie.net halo 3. http://bungie.net, 2010.

[46] C. DeLong, K. Erickson, E. Perrino, K. J. Shim, and N. Pathak. Halofit stats. http://stats.halofit.org, 2009. University of Minnesota, Department of Computer Science, Minneapolis, MN, USA.

[47] K. J. Shim, K. W. Hsu, S. Damania, C. DeLong, and J. Srivastava. An exploratory study of player and team performance in halo 3. In *IEEE Social Computing*, Boston, Massachusetts, USA, 2011.

[48] K. J. Shim, S. Damania, C. DeLong, and J. Srivastava. Player and team performance in everquest ii and halo 3. *IEEE Potentials*, September 2011.

[49] K. J. Shim, N. Pathak, M. A. Ahmad, C. DeLong, Z. Borbora, A. Mahapatra, and J. Srivastava. Analyzing human behavior from multiplayer online game logs: A knowledge discovery approach. *IEEE Intelligent Systems*, 26(1), 2011.

Appendix A

Assembling the MLG Halo 3 dataset

The data used for evaluation for the bulk of this thesis was collected over the course of 2009 as part of a larger project to produce a high-quality, publicly-available competitive gaming dataset [36]. Halo 3, a game released in September 2007 on the Xbox 360 video game console (and the most played online multi-player title on Xbox Live during 2008 and 2009), was played professionally as the flagship game in the Major League Gaming [44] Pro Circuit from 2008 through 2010, as were its predecessors Halo:Combat Evolved and Halo 2 from 2004-2005 and 2006-2007, respectively. Major League Gaming (MLG) is the largest video gaming league in the world and has grown rapidly since its inception in 2003, with Internet viewership for 2009 events topping 750,000.

The dataset contains Halo 3 multi-player games between two teams of four players each. Each game was played in one of two environments - over the Internet on Microsoft's Xbox Live service (commonly referred to as online scrimmages) or on a local area network during MLG tournaments. In both cases, games share a common set of data: the players and teams involved, the date the game was played, which map and game type was played, the overall result and score, and per-player statistics such as kills, deaths, assists (where one player helps another player from the same team kill an opponent), and score (depending on the game type). Each team is assigned a color, red or blue, which help identify players in-game as well as signify the higher seeded team in tournaments

using the color red.

Game types are split between two general categories, Team Slayer and objective. In the Team Slayer game type, players from each team try to kill as many players on the other team as possible. The first team to 50 kills or whomever is ahead after 15 minutes wins the game. Objective game types include Capture the Flag, Oddball, and King of the Hill, the latter two commonly referred to as timed objective games. In the Capture the Flag game type, teams attempt to capture the other's flag with the first to three or five flag captures winning (depending on the map the game is played on). If after 15 minutes teams are tied, the game proceeds to sudden death where the next capture wins. If the game is still tied after 30 minutes, the game is replayed. In timed objective game types, teams will attempt to hold an objective for at most 250 seconds in a 15 minute match. For each second a team holds sole control of the objective they receive a point. In the Oddball variant, teams attempt to obtain and hold a common object (the ball) which can be moved anywhere around the map and will respawn in a central location if not touched by either team for a short period of time. The King of the Hill variant requires teams to occupy a series of fixed locations around the map with only one location considered the "hill" at any given time. After two minutes, the hill moves to another location in a fixed order. If either team does not score 250 points before 15 minutes expire, the team with the most time spent holding the hills wins. In the case of a tie after 15 minutes, the game is replayed.

These games were grouped into series by matches in the case of tournaments (commonly best-of-three or best-of-five matches, which may be extended further if teams meet again later in the double-elimination bracket) or by temporal proximity in the case of online scrimmages. The games wer also kept in the order they were played in the tournament or online scrimmage. For tournament games, the round and bracket (separated between "winners"', "losers"', and "placement" brackets) were tracked for each series and tournaments themselves were grouped into seasons, e.g., the 2008 season, which consisted of several tournaments.

In addition to detailed game data, the dataset contains further information about teams and players. Some Halo 3 teams experienced a high rate of player churn, in some cases replacing all four members of the team, over the course of a season. Figure A.1 depicts the social network formed by shared games between players over the course of

all tournaments contained in the dataset. Players with a majority of games between one another occurring on the same team have a green edge. Those who played a majority of games against one another are red, and players who have spent time as both friend and foe are yellow.

Teams are represented in the dataset as individual entities made up of up to four players. Each change in team status, such as moving from semi-professional to professional status or even complete disbandment are tracked for each team. Team membership is tracked independently of single team or player status, noting when a player joins and leaves any given team. Individual player status is also tracked, such as when one is considered a professional player or moves into retirement. Microsoft's Xbox Live service requires a unique identifier, called a "gamertag", in order for players to log in, and players may have one or more gamertags active at any given time. These gamertags were also kept in the database and assigned a status. Statuses for gamertags include active use, abandoned, renamed, or, in some cases, stolen.

The data made available from these two different environments each required a different method of collection. For the 2008 and 2009 seasons, data from MLG tournaments were provided in a spreadsheet two to three weeks after each tournament concluded. The spreadsheets contained a list of teams playing in the tournament and their seeds, the players on each team, and for each match and each game, the final score and per-player kills, deaths, assists and score. This information was entered into the dataset manually, using a custom interface for quick entry and automated validation of tournament results. This interface also allowed for the management of team, player, and gamertag status changes. Using data from additional resources, such as MLG's web site and online forums, the dataset was kept up-to-date outside of tournament play, which made possible the utilization of automated methods for collecting data for the second environment - online scrimmages conducted over Microsoft's Xbox Live service.

Each game played over Xbox Live was recorded and detailed statistics for it made available on Bungie's web site, bungie.net [45]. In addition, players could view their personal profiles listing summary statistics about their performance and a list of each game they played online. Utilizing information collected about players and their associated gamertags, a spider was created to iterate over all active players looking for games where all eight players were known and the game exhibited a tournament-like configuration,

i.e., the maps and game types corresponded to tournament settings. Approximately every 30 minutes, this spider would examine the game history of each player searching for games they shared in common with their teammates. If it found a shared game, it would then retrieve statistics for that particular game if the other team could be identified as a known team consisting of four professional players and none of the eight players had prematurely disconnected from the game. These games were then added to a verification queue. In most cases, games were automatically verified if they were from a known map and game type and there was a winning score, i.e., the game concluded successfully. Games that could not be automatically verified were checked by hand. The crawler was later extended to help find additional gamertags by flagging games for review where it had identified 7 of 8 players. This allowed for the semi-automated detection of team changes and player try-outs before they were reported on MLG's web site or forums.

The dataset has been made available on the Project HaloFit web site in two forms. The first, HaloFit Stats [46], contains several views into the dataset similar to statistics pages of professional sports leagues such as Major League Baseball or the National Hockey League. Users can drill down into the dataset using a series of filters to find data relevant to favorite teams or players. Figure A.2 shows the team detail page for the team "Carbon" using several filters to display the team's performance against the team "Str8 Rippin" in 2009 tournaments. The second, the Project HaloFit Data Repository [36], contains partial and full comma-separated exports of the dataset, as well as detailed documentation. The dataset currently houses information on over 9,100 games, 566 players, and 186 teams.

Figures A.3 and A.4 illustrate the social network formed by players scrimmaging before and playing during the 2009 MLG Anaheim tournament, respectively. Each node represents a player and the edges between nodes denote shared games. Team names have been overlaid as labels, and correspond to the team membership for a given group of 4 players. The edges were weighted by the number of games shared between the two players. The graph was created using a force-directed layout where the nodes repulse one another linearly based on proximity and the edges pull the nodes together with a force logarithmically proportional to their weight. In both graphs, the edge width is also an indication of the assigned weight.

Four-player teams who played together often appear as tight clusters of four nodes, while teams that had player turnover or did not practice as often appear loose, lopsided, or contain more than four nodes (in cases were teams split or held try-outs for open positions). Teams that played frequently with other teams are closer to each other, while teams that played sparingly against other teams are further apart. Figure A.3 shows a tight core of higher-seeded teams with some lower-seeded teams around the core and moving further into the periphery. In this example, the graph is not completely connected. The disconnected cluster is comprised of two European teams who only scrimmaged against each other before the tournament.

Figure A.4 details the network formed during the tournament and highlights the asymmetrical double-elimination structure of the tournament. The legs are the lower-seeded teams who started well back in the losers' bracket, while the core consists of teams who started higher in the losers' bracket or who started in the winners' bracket. In this particular example, teams who are found in the core of the scrimmage graph are largely found in the middle of the tournament graph, and those who are found in the periphery of the scrimmage graph are found on the outside edges or legs of the tournament graph.

The dataset has several interesting characteristics, such as the high frequency of team changes from one tournament to the next. With four players per team, it is not uncommon for a team with a poor showing in one tournament to replace one or two players before the next. As such, the resulting dataset lends itself to analyses of skill at the subgroup level because the diversity of player assignments can aid in isolating interesting characteristics of teams who do well versus those who do not. Additionally, since the players making up the top professional and semi-professional teams are all highly-skilled individually, basic game familiarity (such as control mechanics) are not considered as important a factor in winning/losing as overall team strategy, execution, and adaptation to the opposition in real-time. This focus also helps mitigate issues pertaining to the personal motivations of players since all must be dedicated to winning in order to have earned a spot in the top 32 teams in the league, winnowing out those who might intentionally lose games for their teams (as is commonplace in typical Halo 3 multi-player gaming). Taken together, these elements make for a very high quality research dataset for those interested in studying competitive gaming, skill ratings systems,

and teamwork (several analyses have already been conducted [47], [48], [49]).

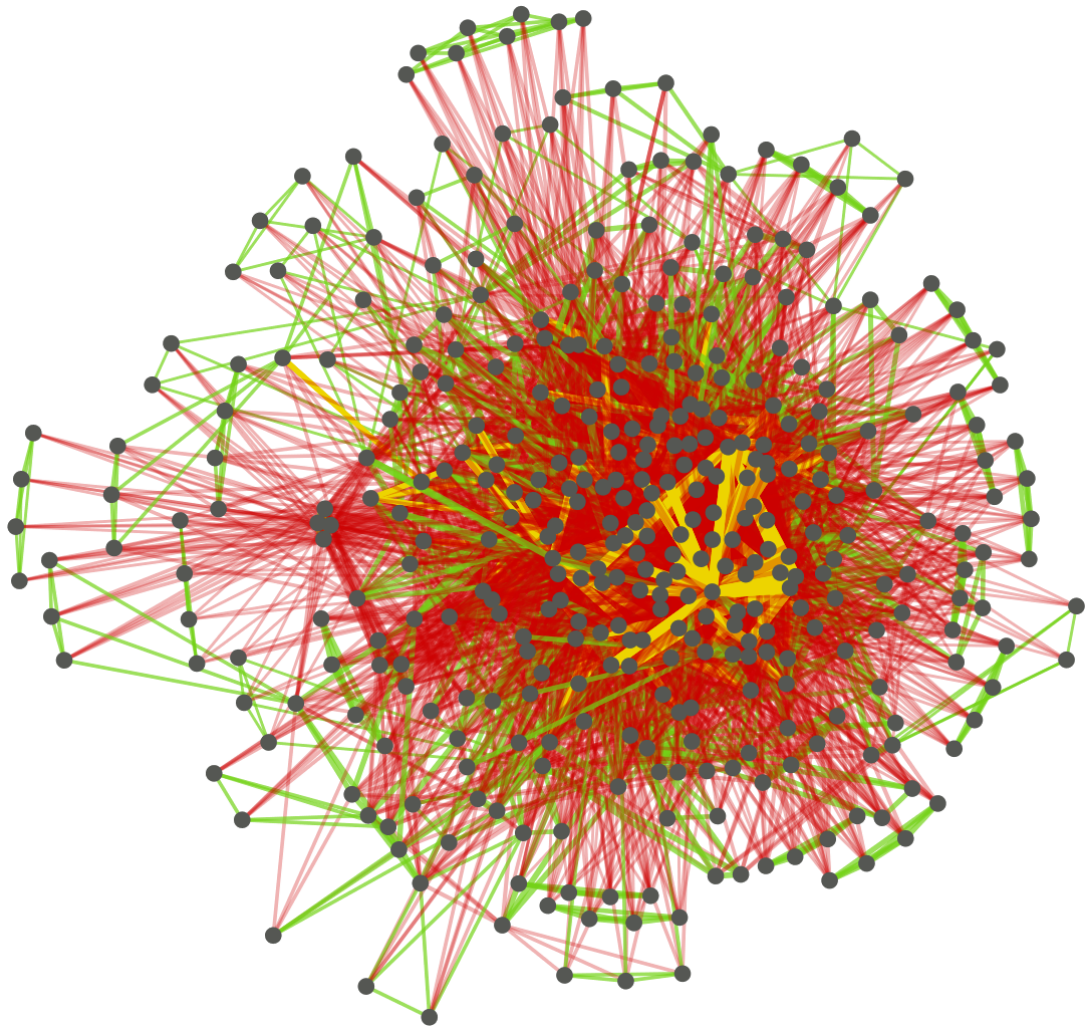


Figure A.1: A visualization of the "friend or foe" social network for all tournaments in 2008 and 2009.

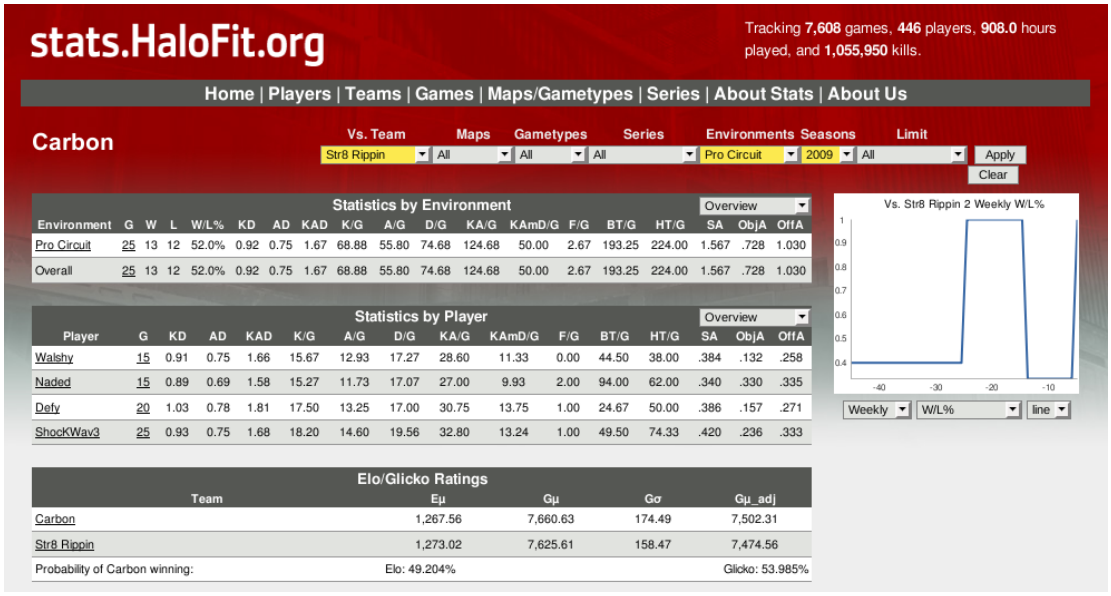


Figure A.2: The Project HaloFit web site showing a team detail page for Carbon.

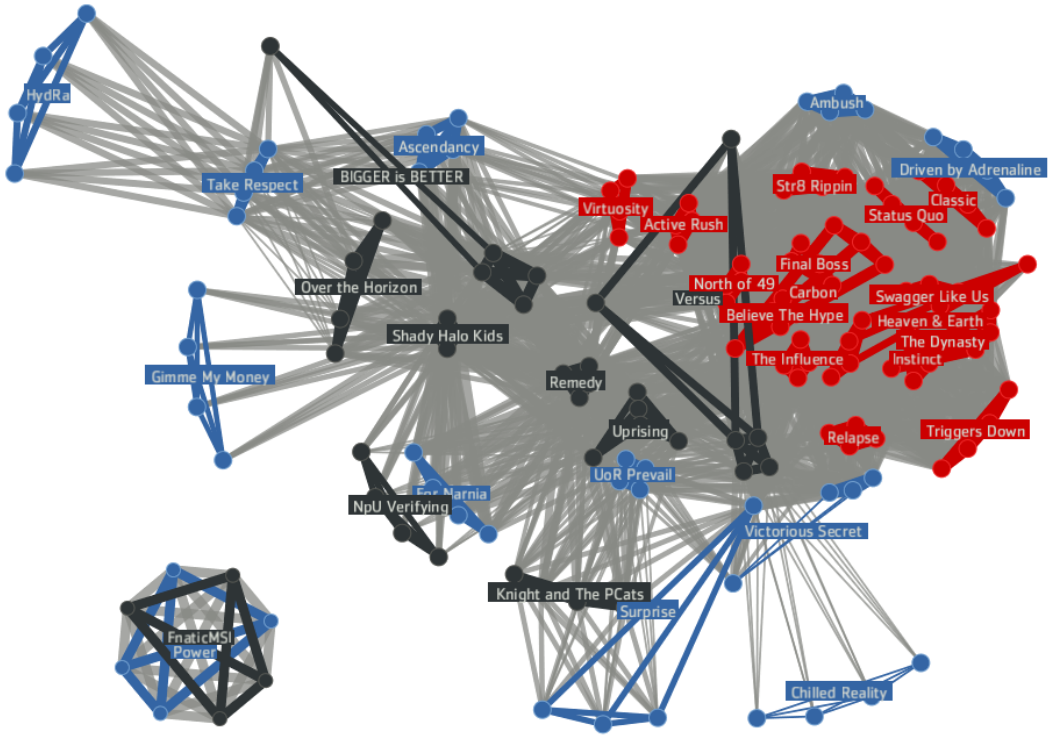


Figure A.3: A visualization of the social network among teams based on online scrimmage data prior to the Anaheim 2009 tournament.

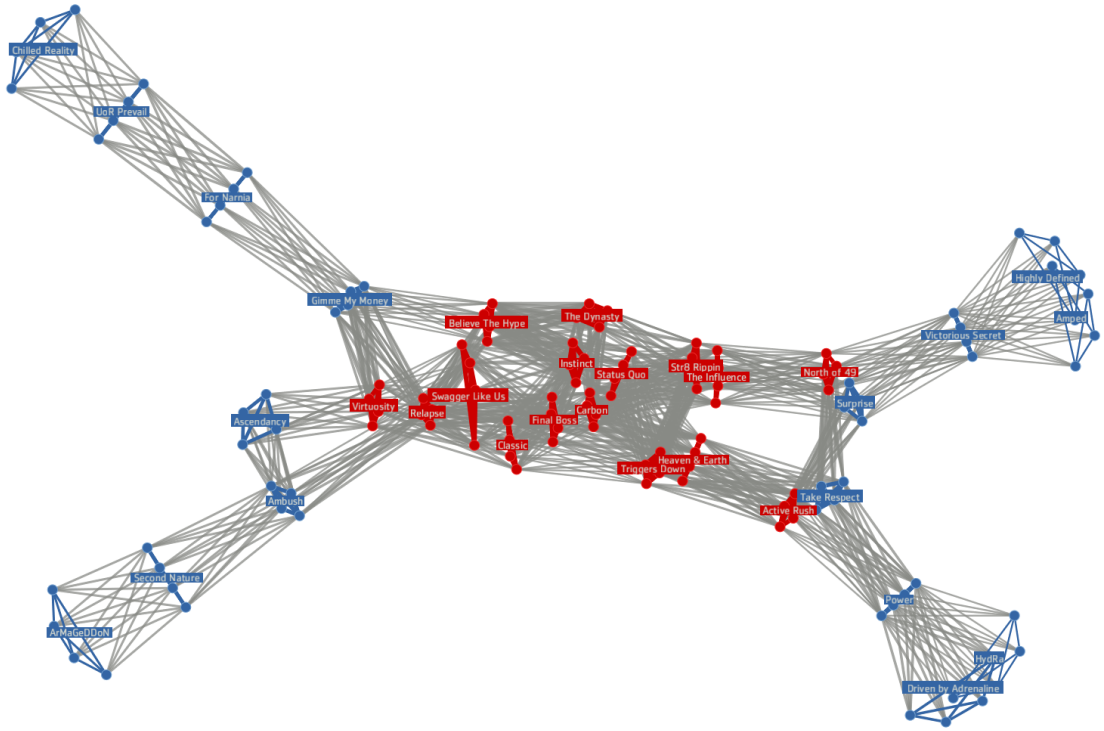


Figure A.4: A visualization of the social network among the 32 teams playing in the Anaheim 2009 tournament.

Appendix B

Assembling the NBA dataset

In Chapter 6, we evaluate the TeamSkill approaches using a dataset comprised of basketball games played in the National Basketball Association (NBA) during the 2011-2012 regular season. However, locating a dataset from real-world team-based sports at the level of granularity necessary for this work was a non-trivial endeavor.

A key advantage of the MLG Halo 3 dataset is that players are not substituted while the game is being played - a characteristic common to the vast majority of real-world team-based sports. Injuries or personal fouls may necessitate player substitution and so games are essentially a series of competitions between subsets of players from each team. The skill assessment techniques described in this thesis, including the TeamSkill suite and all the baseline approaches, were designed to work at the level of games and implicitly assume the rankable entity - be they players or subgroups - was present at all points in time during the game. This assumption is valid in most online multi-player video games, and clearly for one-on-one competitions, and is what allows us to say that player/subgroup ratings are a function of their past opponents' skill. If there can be multiple configurations of opponents (and teammates) within a single game, this assumption is no longer valid and, as such, the ratings can no longer function at the level of the entire game. Instead, we must break the game into segments where this assumption is once again valid, i.e., when the team configuration is constant for some period of time.

The task before us, then, was to find a real-world team-based sport in which there existed data available at this granularity, at least for the data elements assumed by the

general case 1.2.2 and, ideally, game-specific variables to evaluate EVGen and EVMixed. Unfortunately, after much searching, we found this too high a bar for nearly all publicly-available datasets, of which there were few to choose from to begin with. It seems likely that this dearth of fine-grained publicly-available data for sports is due, in part, to the prevalence of ad-supported (or otherwise commercialized) statistic aggregation websites.

One candidate collection of datasets was found, though. At the time of writing, Basketballvalue.com [41] offers several data files for NBA games played during the 2005 through 2011 regular seasons, as well as several data files for each season's playoffs. For each season or post-season playoffs, there are 8 data files [1] :

- List of all games in the season
- Play by play of all games in the season
- List of each matchup of one unit against another
- List of aggregated matchups of one unit against another
- List of players and player IDs
- Statistics of all players across all teams played for
- Statistics of all players broken down by team played for
- Statistics of all teams

Of these data files, only 2 are of use for the skill assessment task described above: "Play by play of all games in the season" and "List of each matchup of one unit against another".

The play-by-play file contains a list of actions occurring in each game, each action's timestamp within the game, and a text entry describing the action (e.g., turnover, 3pt shot, foul, etc). No other real-world team-based sports dataset was found with this level of detail. A sample of the play-by-play data is given in Table B.1.

The other file is "List of each matchup of one unit against another". "Match-ups" are an instance of a configuration of players from a team and their opposition who compete for some duration of time until a boundary condition, such as a player substitution or

[1] For consistency, data file descriptions are taken from [41]

Table B.1: Sample data from the 2011-2012 NBA data file "Play by play of all games in the season".

GameID	LineNumber	TimeRemaining	Entry
20111225BOSNYK	1	0:48:00	Jump Ball Chandler vs O'Neal (Rondo gains possession)
20111225BOSNYK	2	0:47:41	[BOS] Allen Turnover : Lost Ball (1 TO) Steal:Fields (1 ST)
20111225BOSNYK	3	0:47:18	[NYK] Anthony Turnover : Bad Pass (1 TO) Steal:Rondo (1 ST)
20111225BOSNYK	4	0:47:10	[NYK] Douglas Foul: Shooting (1 PF)
20111225BOSNYK	5	0:47:10	[BOS 1-0] Rondo Free Throw 1 of 2 (1 PTS)
20111225BOSNYK	6	0:47:10	[BOS 2-0] Rondo Free Throw 2 of 2 (2 PTS)
20111225BOSNYK	7	0:46:57	[NYK 3-2] Douglas 3pt Shot: Made (3 PTS) Assist: Fields (1 AST)
20111225BOSNYK	8	0:46:34	[BOS] Garnett Jump Shot: Missed
20111225BOSNYK	9	0:46:32	[NYK] Anthony Rebound (Off:0 Def:1)
20111225BOSNYK	10	0:46:25	[BOS] O'Neal Foul: Shooting (1 PF)
20111225BOSNYK	11	0:46:25	[NYK 4-2] Chandler Free Throw 1 of 2 (1 PTS)
20111225BOSNYK	12	0:46:25	[NYK] Chandler Free Throw 2 of 2 Missed
20111225BOSNYK	13	0:46:23	[NYK] Anthony Rebound (Off:1 Def:1)
20111225BOSNYK	14	0:46:13	[NYK 6-2] Douglas Running Jump Shot: Made (5 PTS)
20111225BOSNYK	15	0:46:03	[BOS 4-6] Garnett Layup Shot: Made (2 PTS) Assist: Rondo (1 AST)
20111225BOSNYK	16	0:45:46	[NYK] Anthony Jump Shot: Missed
20111225BOSNYK	17	0:45:45	[BOS] O'Neal Rebound (Off:0 Def:1)

Table B.2: Sample data from the 2011-2012 NBA data file "List of each matchup of one unit against another". The last 3 columns are stand-ins for the larger list of home/away team players and the home/away statistics, such as points scored or rebounds, for a particular match-up.

GameID	StartTime	EndTime	Player$i$..M	Player$j$..N	Statistic1..Z
20111225BOSNYK	0:48:00	0:42:59	*	*	*
20111225BOSNYK	0:42:59	0:42:17	*	*	*
20111225BOSNYK	0:42:17	0:41:42	*	*	*
20111225BOSNYK	0:41:42	0:40:35	*	*	*
20111225BOSNYK	0:40:35	0:39:37	*	*	*
20111225BOSNYK	0:39:37	0:39:02	*	*	*

half-time, is reached, at which point a new match-up begins. An NBA game consists of many such match-ups, and the score of each team is the sum of each team's score from the set of all match-ups for that game. Here, the matchup file "List of each matchup of one unit against another" contains a list of all the match-ups occurring in each game, the starting and ending timestamp for each match up within the game, the players on each team, and a number of statistics summarizing the performance of each team's players for that match-up. A sample of the match-up data is given in Table B.2.

Table B.3 summarizes some of the key metrics regarding the NBA dataset. Many match-ups tend to be short (relative to the lengths of Halo games) with an average length of 101.33 seconds, with the longest observed being 720 seconds. In figure B.1, a

Table B.3: Summary statistics for the 2011-2012 NBA regular season dataset

Number of games recorded	883
Number of teams recorded	30
Number of players recorded	474
Average number of match-ups per game	28.65
Average length of match-up (seconds)	101.33

histogram of match-up lengths for the dataset is provided, which underscores the fact that most NBA match-ups are 80 seconds or less. Only about 20% are longer than 150 seconds.

For the purposes of evaluating the TeamSkill and baseline approaches, the match-up dataset B.2 is nearly enough detail alone. Unfortunately the match-up statistics only extend to the level of each team, not each player for each team, the latter of which is required for EVGen and EVMixed. As such, the per-player statistics from the play by play dataset B.1 had to be extracted and merged with the information from the match-up dataset before they could be processed by the TeamSkill or baseline approaches.

Other data cleaning and validation tasks were required as well as there were a number of problems with the source data files. The play by play data file B.1 had to be resorted according to "GameID", "LineNumber", and "TimeRemaining" since it was discovered that "LineNumber" values weren't always in ascending order, especially in cases where multiple actions occurred within a single time instance, e.g., multiple player substitutions and a timeout. In other cases, the end time boundary for a match-up would not correspond with the first boundary condition found within that match-up in the play by play data, such as finding a player substitution with a timestamp occurring prior to the end time boundary from the match-up data file. There were also cases where entire games were missing from the match-up data, but present in the play by play data, and so the play by play file had to be "fast forwarded" until the next shared match-up was found. The end result of this process was a single, cleaned dataset having one row of data per player per team per match-up per game.

In comparison to the MLG Halo 3 dataset, where team configurations are constant during games and may change after tournaments, in the NBA, the teams are constant throughout a season (for the most part) and the configurations change within games themselves, i.e., match-ups. As such, the NBA bears some resemblance to Halo in that

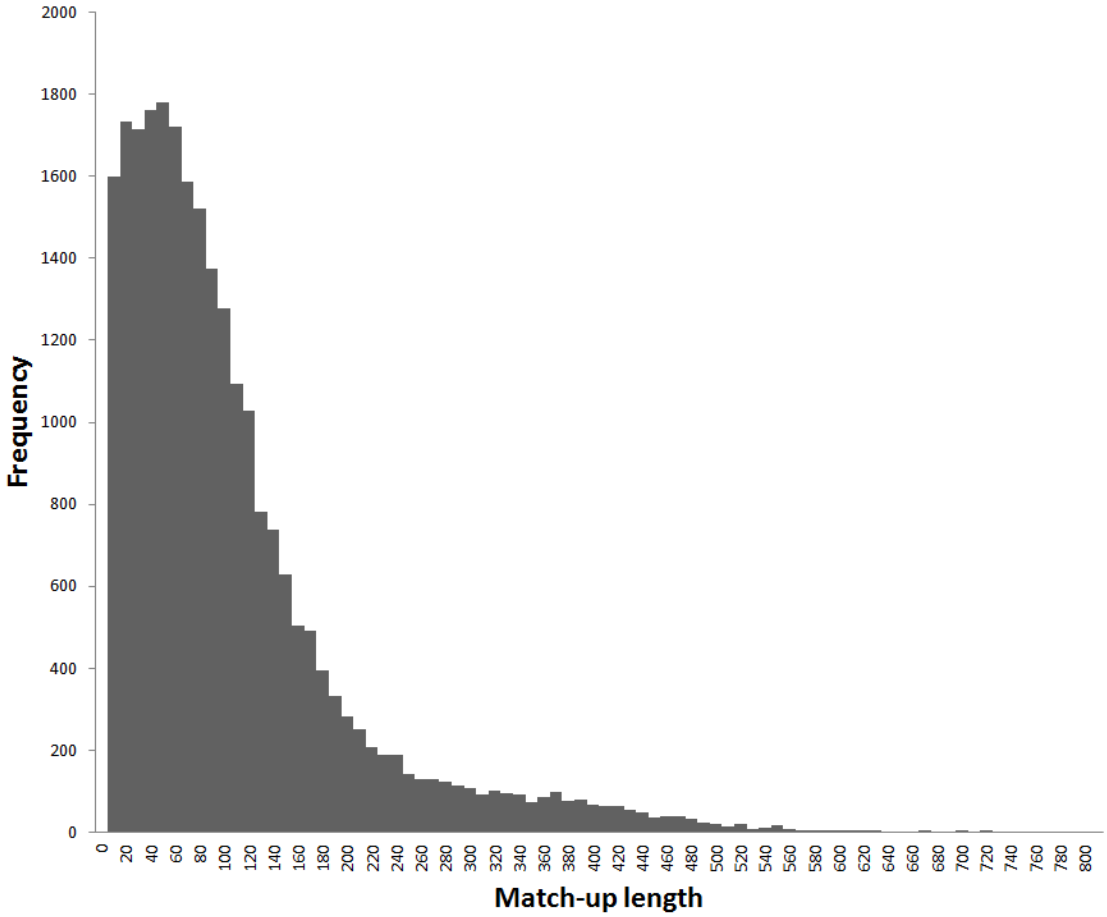


Figure B.1: Match-up length histogram for the 2011-2012 NBA regular season dataset (in seconds).

configurations may change, albeit in different contexts. With respect to the TeamSkill approaches, it is this constantly-changing set of team configurations (in the form of match-ups) which provides clues as to the chemistry of subgroups of players on a team. This makes for an interesting data set for team chemistry research, and a real world team-based sports analogue to the virtual world of professional Halo.

Appendix C

Complete TeamSkill-K, AllK, AllK-EV, and AllK-LS evaluation results

C.1 Complete history - all data before the test tournament

C.2 Recent history - all data between the test tournament and the one preceding it

C.3 Long history - all data except for the data between the test tournament and the one preceding it

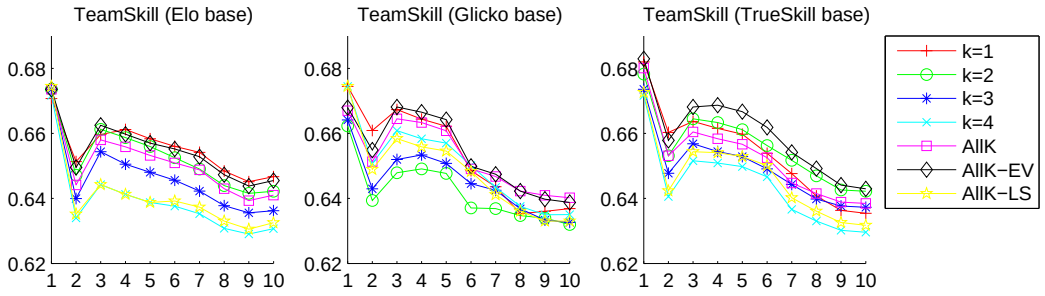


Figure C.1: Prediction accuracy for both tournament and scrimmage/custom games using complete history.

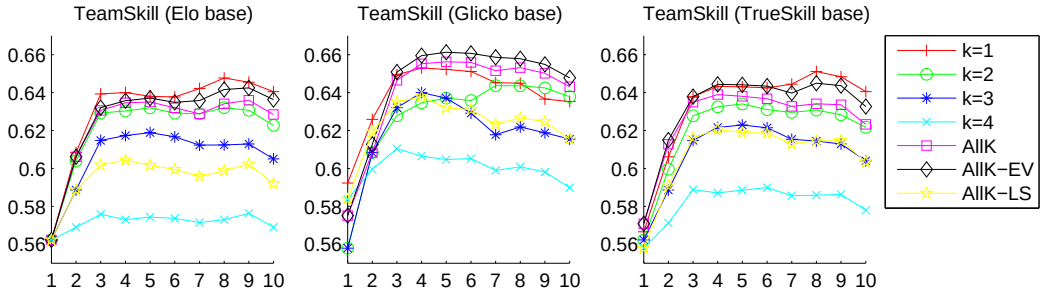


Figure C.2: Prediction accuracy for tournament games using complete history.

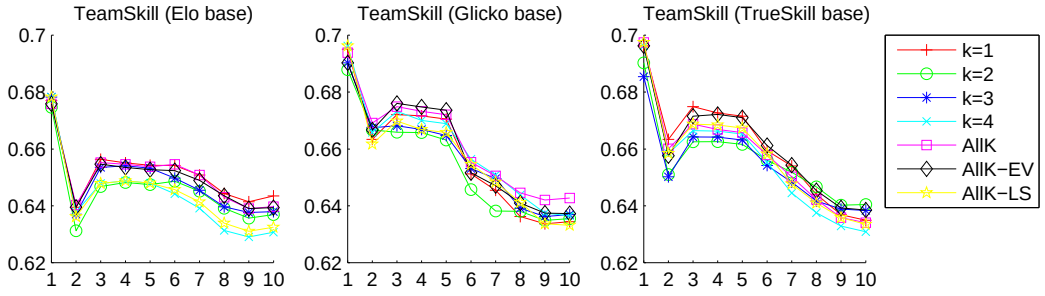


Figure C.3: Prediction accuracy for scrimmage/custom games using complete history.

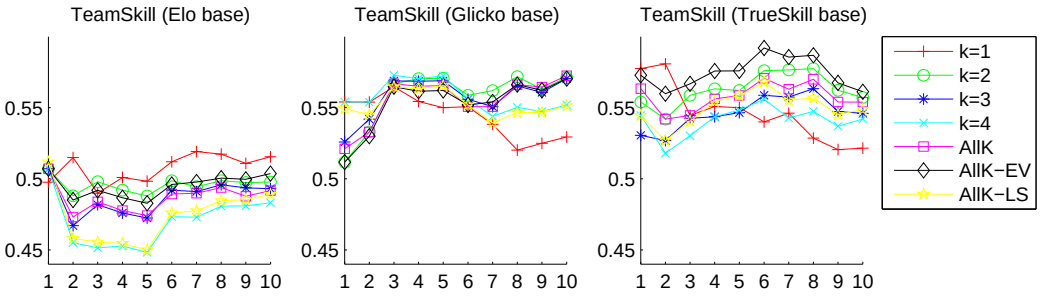


Figure C.4: Prediction accuracy for both tournament and scrimmage/custom games using complete history, close games only.

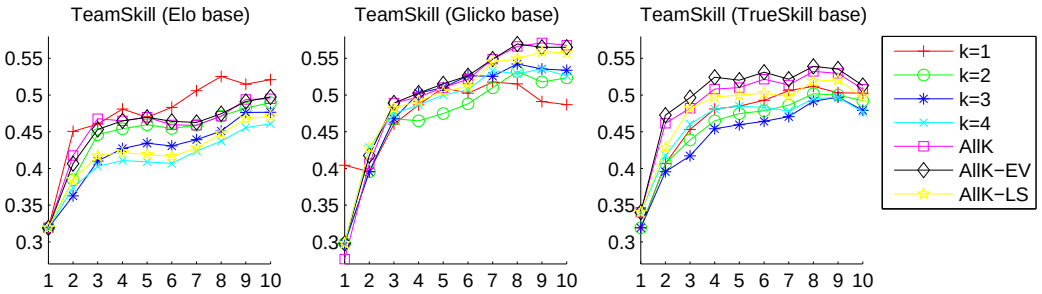


Figure C.5: Prediction accuracy for tournament games using complete history, close games only.

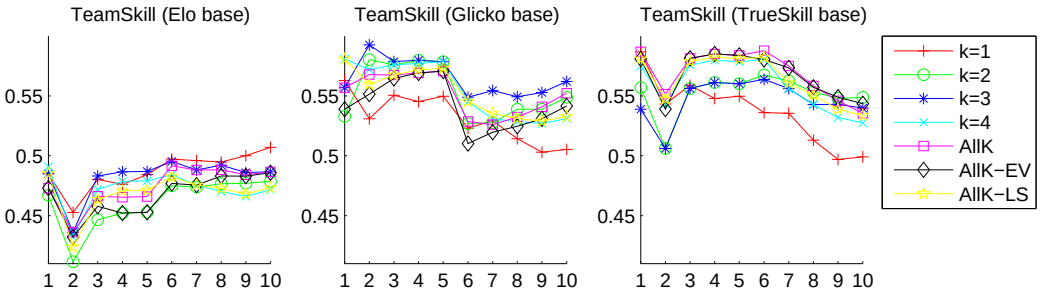


Figure C.6: Prediction accuracy for scrimmage/custom games using complete history, close games only.

Figure C.7: Prediction accuracy for both tournament and scrimmage/custom games using recent history.

Figure C.8: Prediction accuracy for tournament games using recent history.

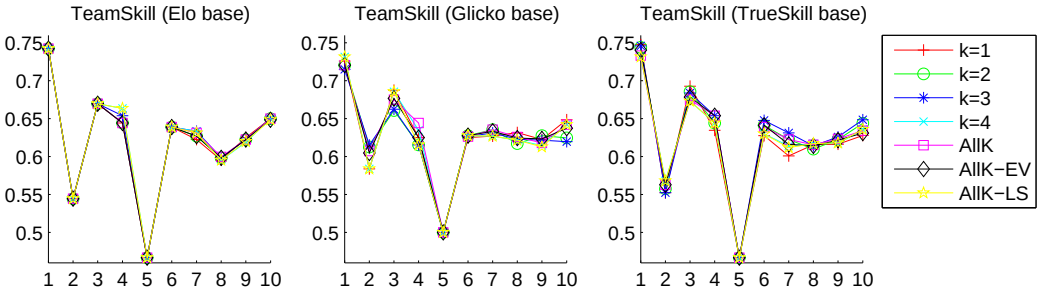


Figure C.9: Prediction accuracy for scrimmage/custom games using recent history.

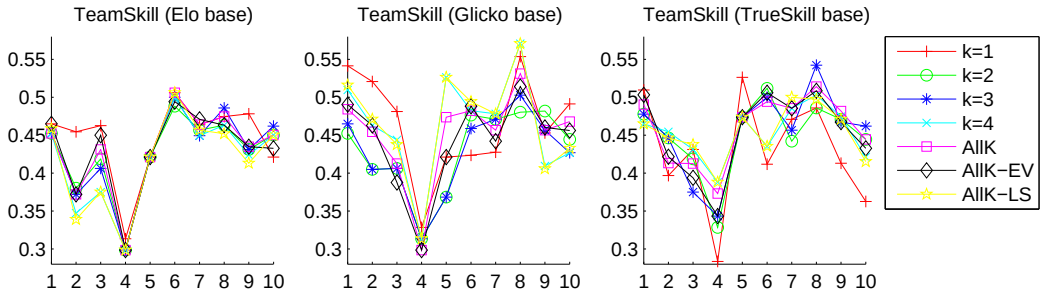


Figure C.10: Prediction accuracy for both tournament and scrimmage/custom games using recent history, close games only.

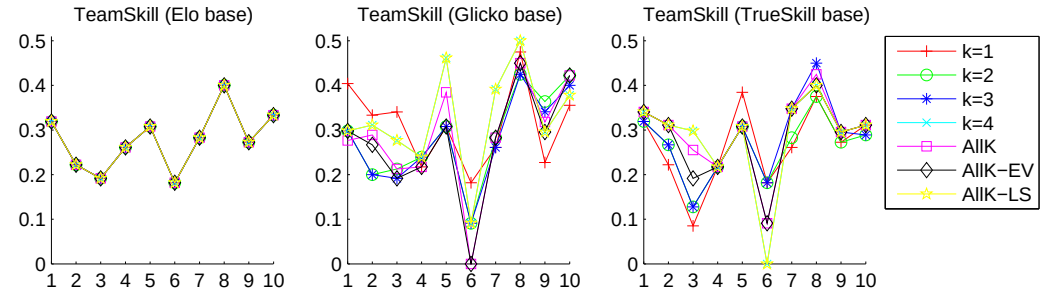


Figure C.11: Prediction accuracy for tournament games using recent history, close games only.

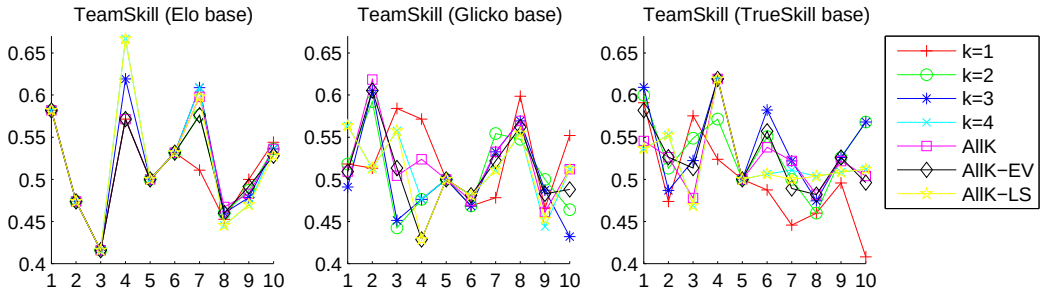


Figure C.12: Prediction accuracy for scrimmage/custom games using recent history, close games only.

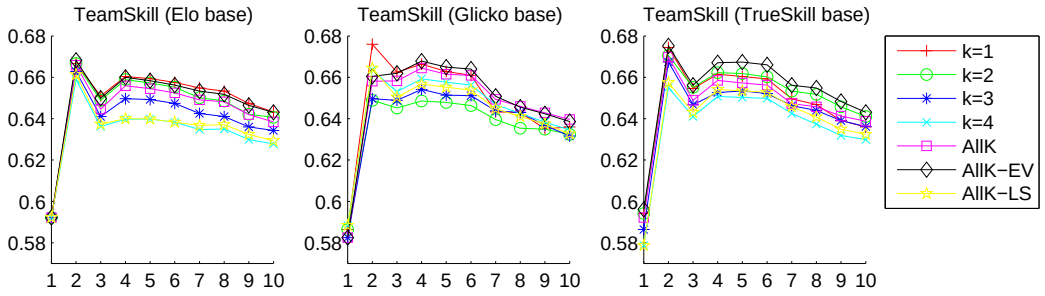


Figure C.13: Prediction accuracy for both tournament and scrimmage/custom games using long history.

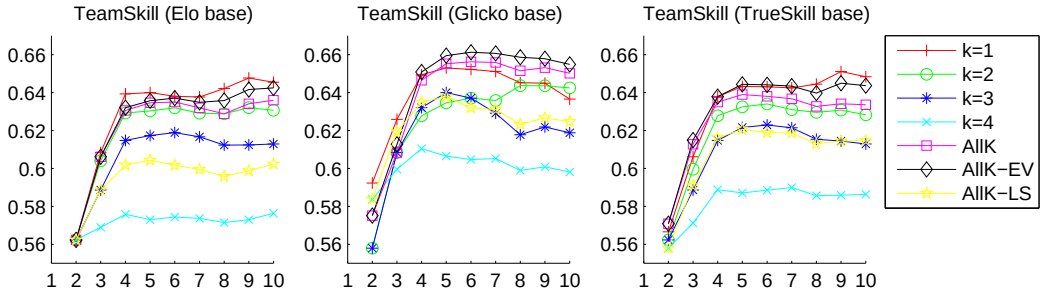


Figure C.14: Prediction accuracy for tournament games using long history.

Figure C.15: Prediction accuracy for scrimmage/custom games using long history.

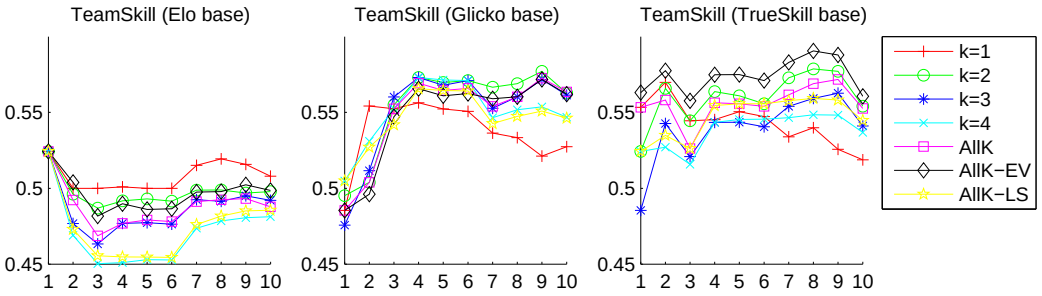


Figure C.16: Prediction accuracy for both tournament and scrimmage/custom games using long history, close games only.

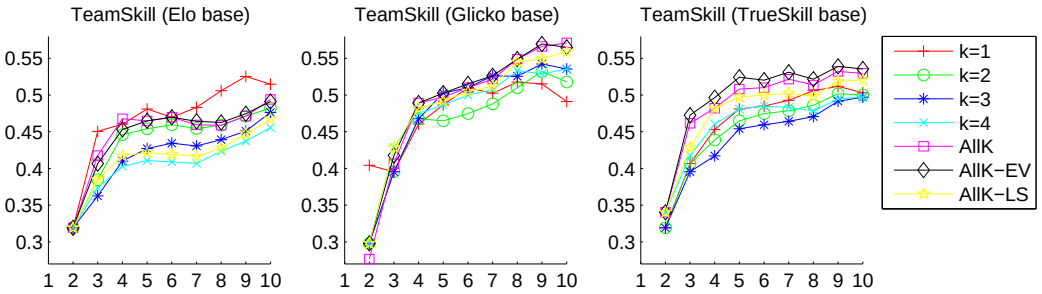


Figure C.17: Prediction accuracy for tournament games using long history, close games only.

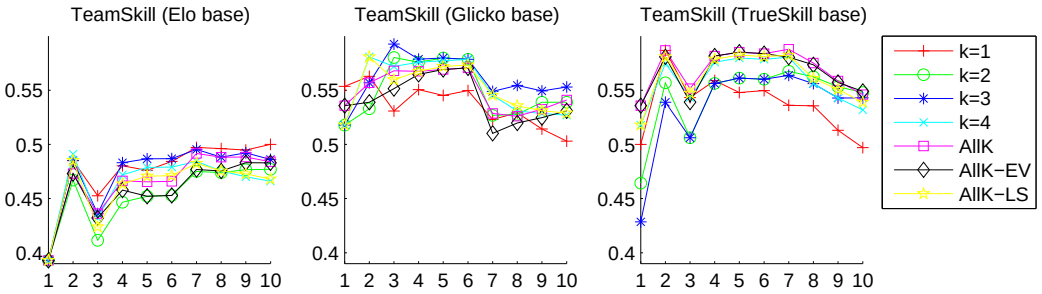


Figure C.18: Prediction accuracy for scrimmage/custom games using long history, close games only.

Appendix D

Complete TeamSkill-AllK-EV-OL1/OL2/OL3, EVGen, and EVMixed evaluation results

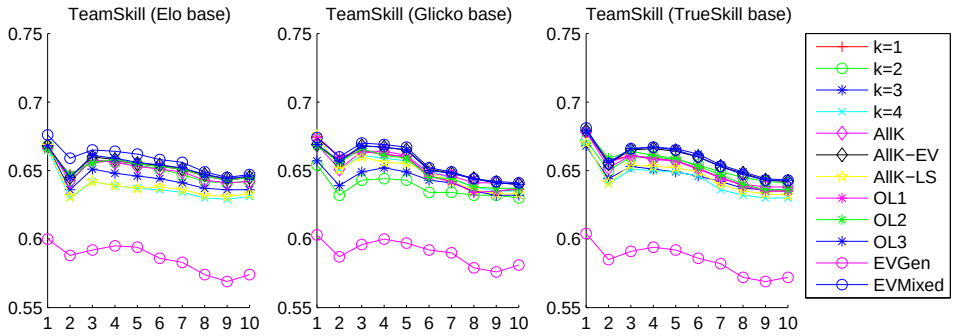


Figure D.1: Prediction accuracy for both tournament and scrimmage/custom games using complete history.

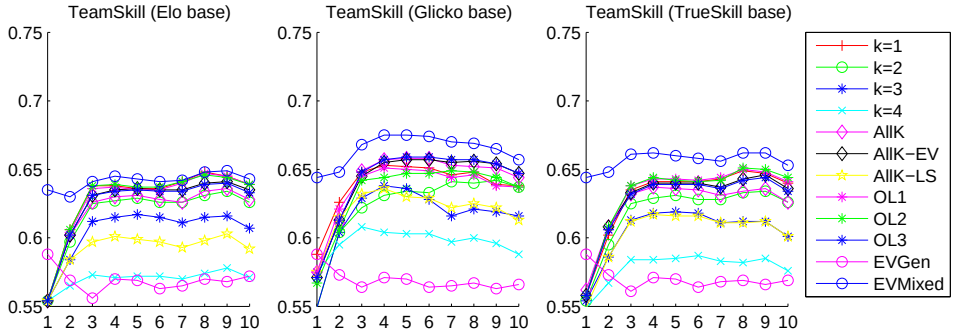


Figure D.2: Prediction accuracy for tournament games using complete history.

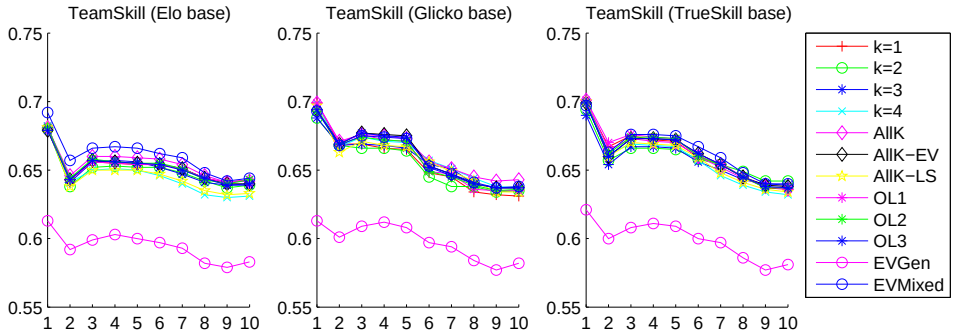


Figure D.3: Prediction accuracy for scrimmage/custom games using complete history.

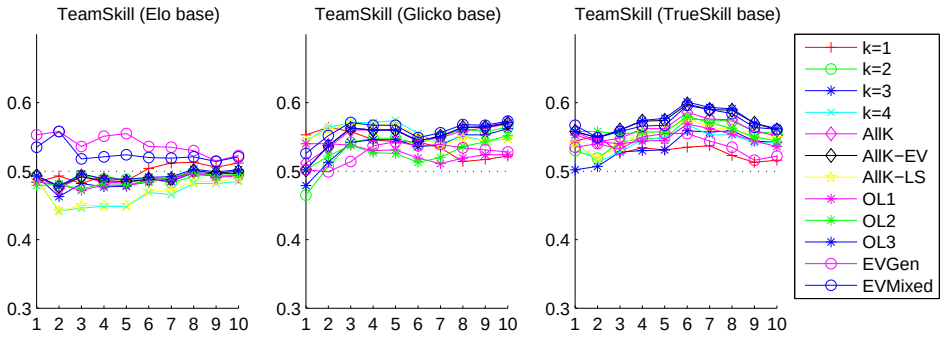


Figure D.4: Prediction accuracy for both tournament and scrimmage/custom games using complete history, close games only.

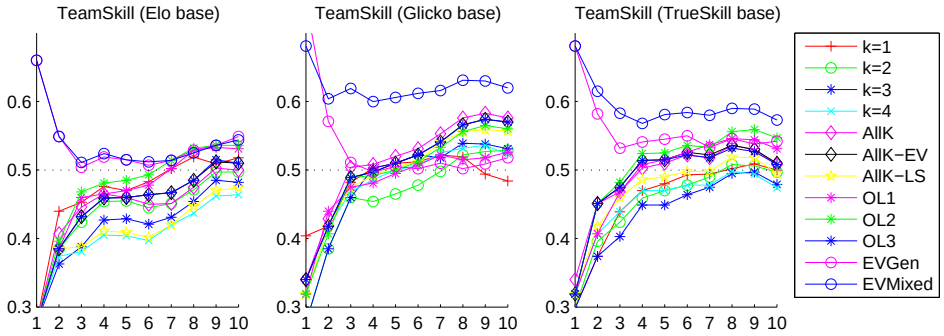


Figure D.5: Prediction accuracy for tournament games using complete history, close games only.

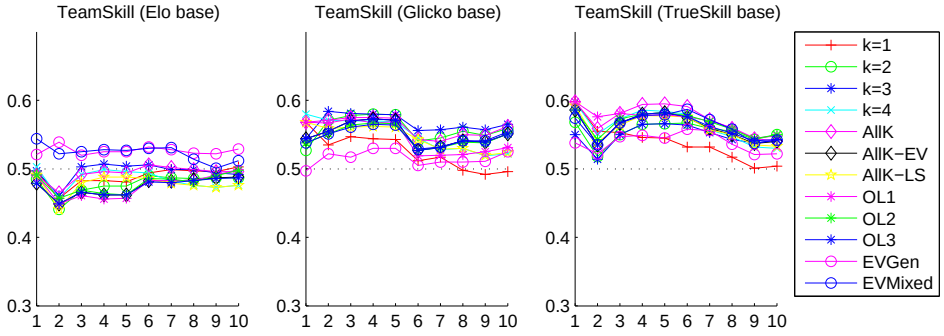


Figure D.6: Prediction accuracy for scrimmage/custom games using complete history, close games only.

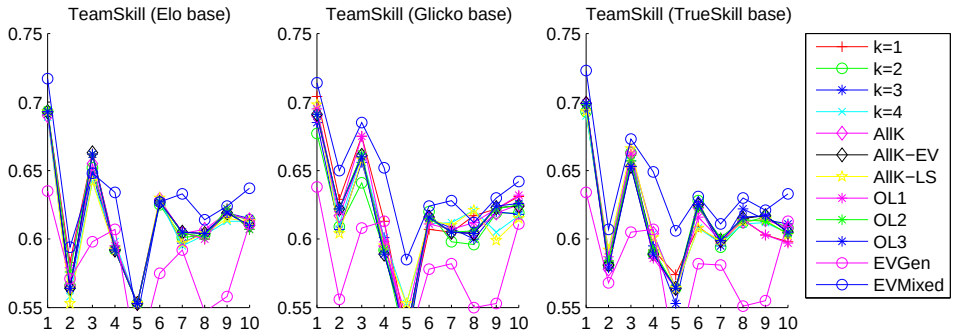


Figure D.7: Prediction accuracy for both tournament and scrimmage/custom games using recent history.

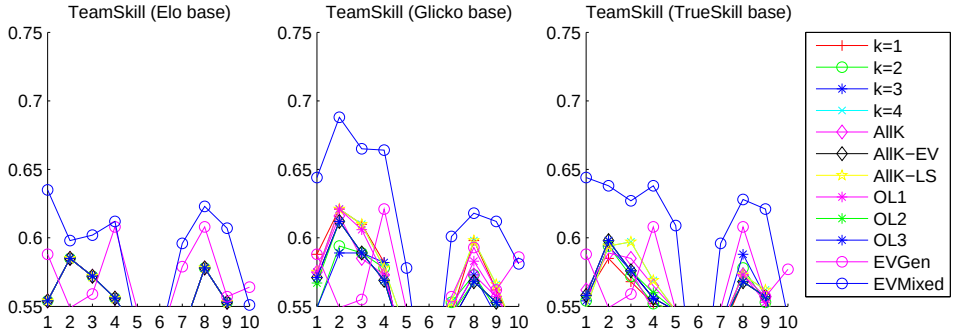


Figure D.8: Prediction accuracy for tournament games using recent history.

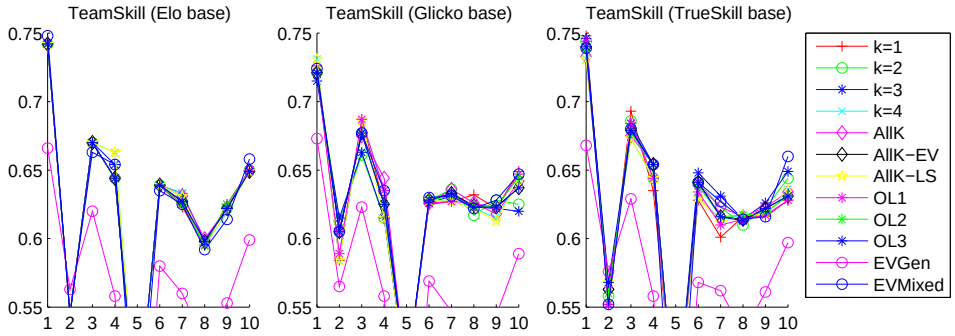


Figure D.9: Prediction accuracy for scrimmage/custom games using recent history.

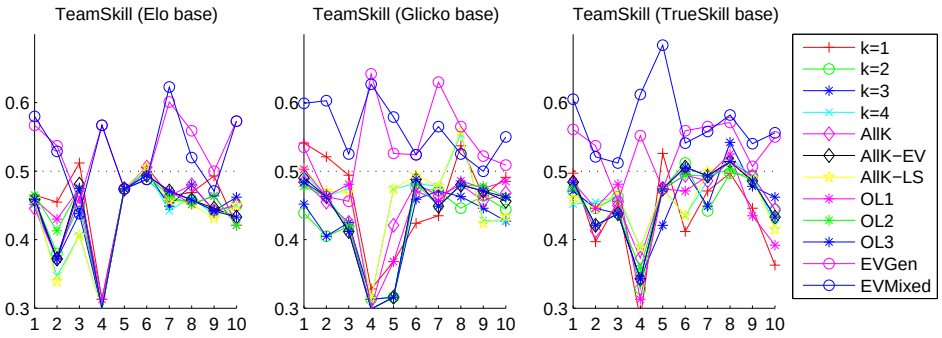


Figure D.10: Prediction accuracy for both tournament and scrimmage/custom games using recent history, close games only.

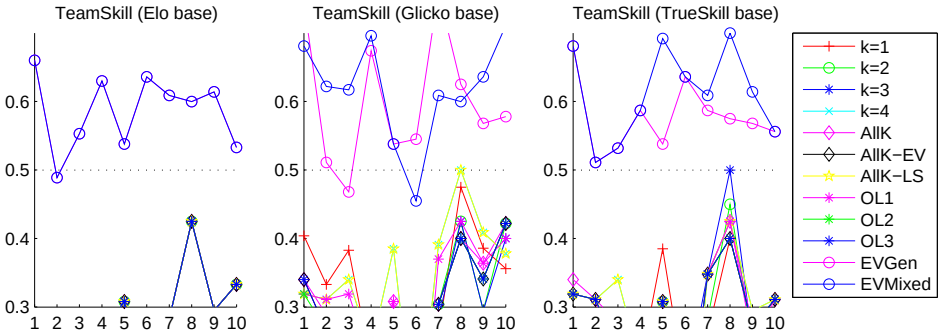


Figure D.11: Prediction accuracy for tournament games using recent history, close games only.

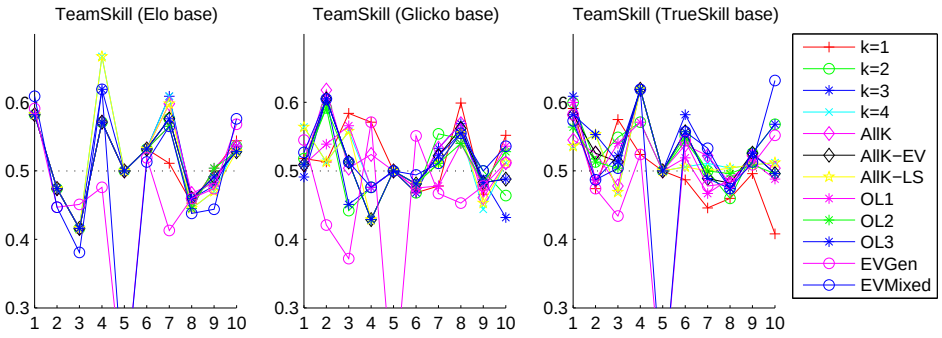


Figure D.12: Prediction accuracy for scrimmage/custom games using recent history, close games only.

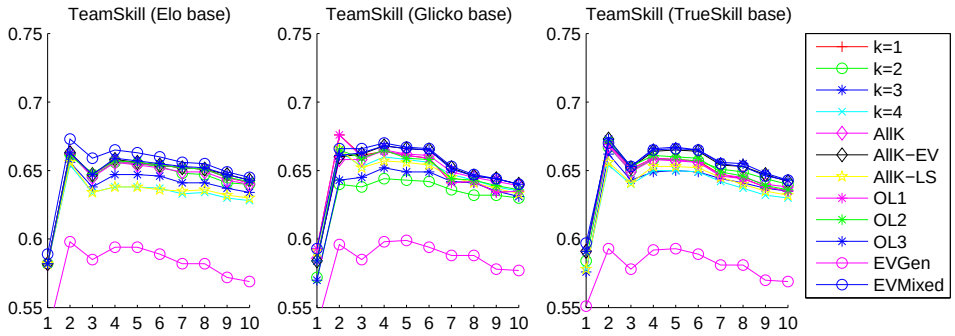


Figure D.13: Prediction accuracy for both tournament and scrimmage/custom games using long history.

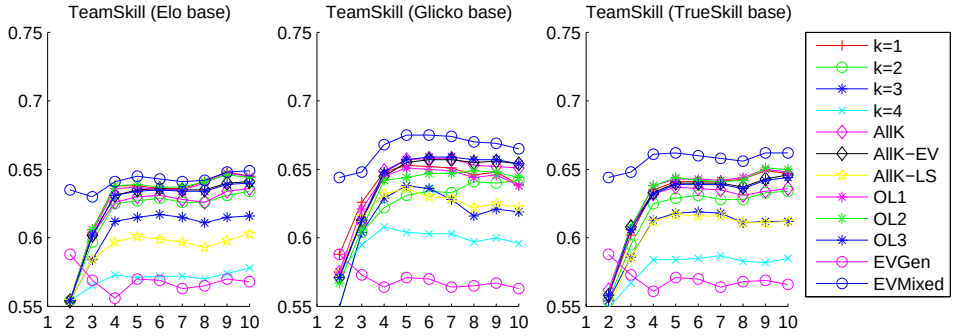


Figure D.14: Prediction accuracy for tournament games using long history.

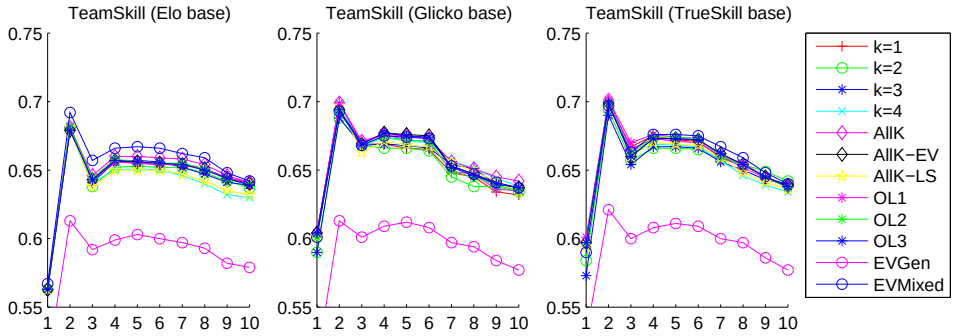


Figure D.15: Prediction accuracy for scrimmage/custom games using long history.

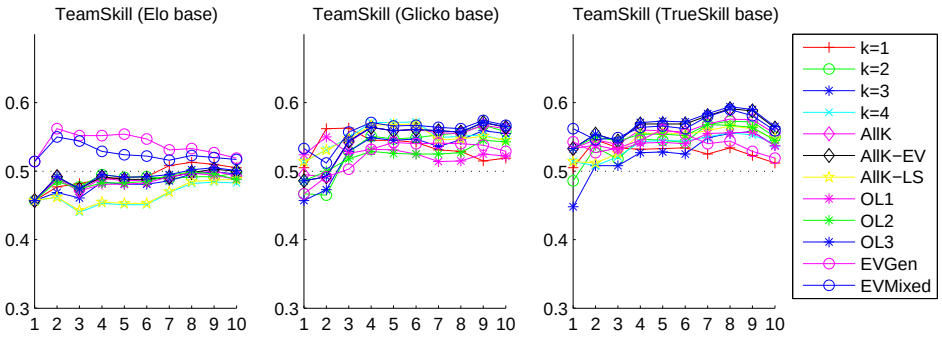


Figure D.16: Prediction accuracy for both tournament and scrimmage/custom games using long history, close games only.

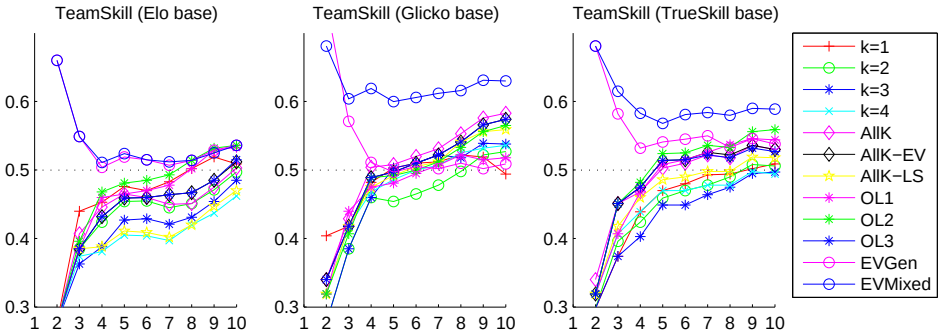


Figure D.17: Prediction accuracy for tournament games using long history, close games only.

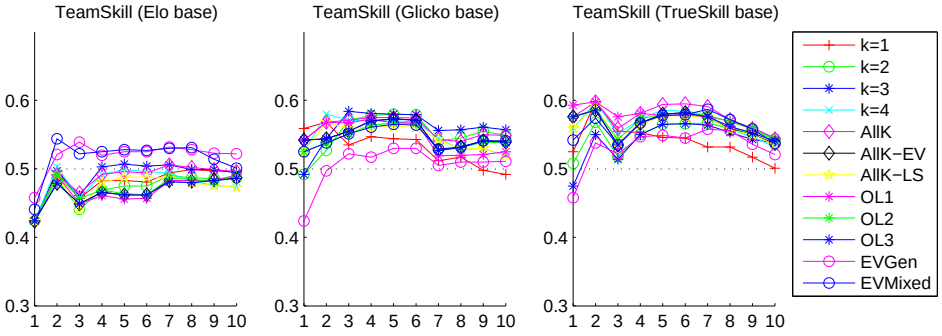


Figure D.18: Prediction accuracy for scrimmage/custom games using long history, close games only.

Appendix E

Final skill distributions for TeamSkill-K using the MLG Halo 3 dataset

E.1 All tournament and scrimmage games - complete history

E.2 All tournament games - complete history

E.3 All scrimmage games - complete history

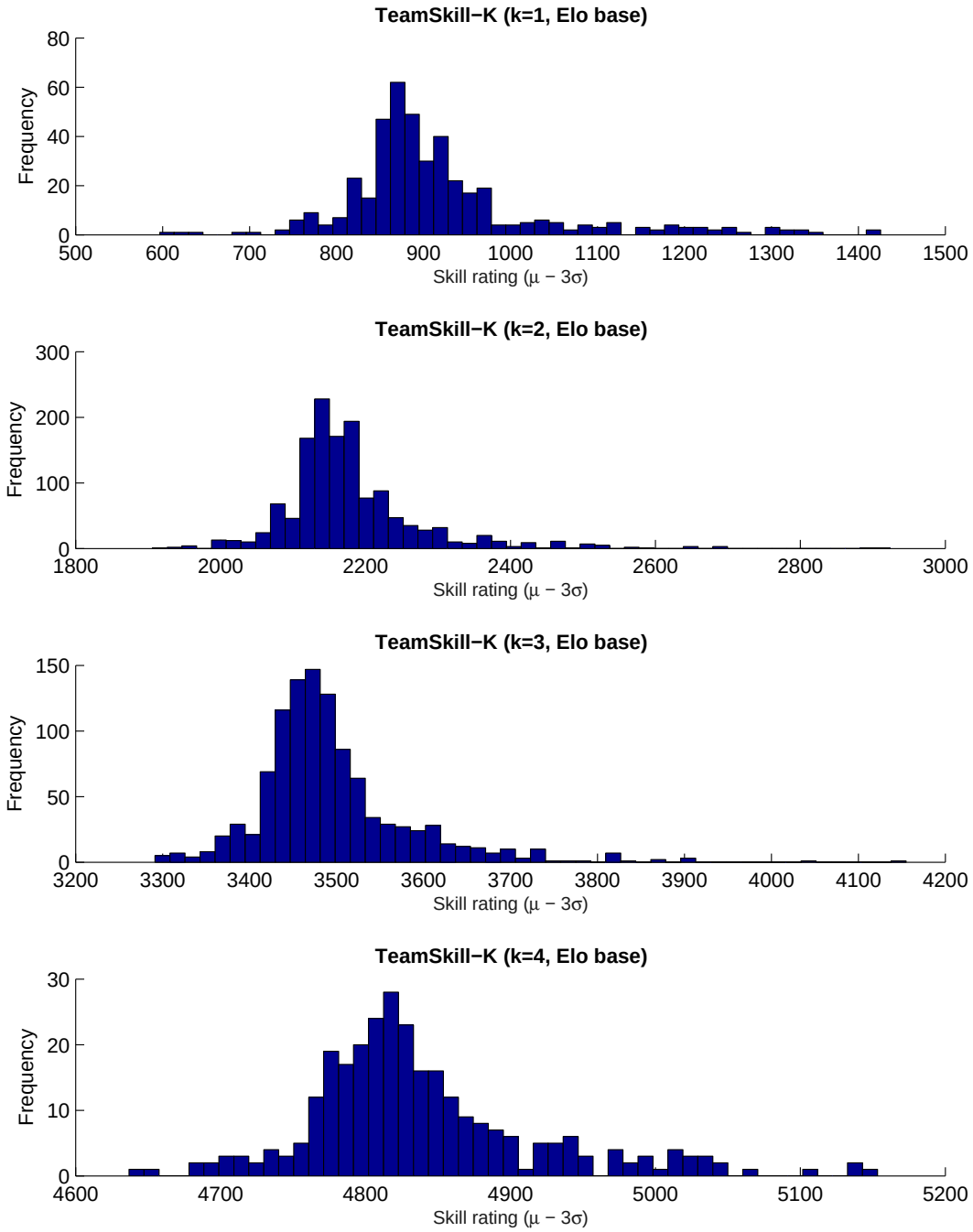


Figure E.1: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with Elo ($\beta = 193.4364$, $\mu_0 = 1500k$, $\sigma_0^2 = \beta^2 k$) as the base learner.

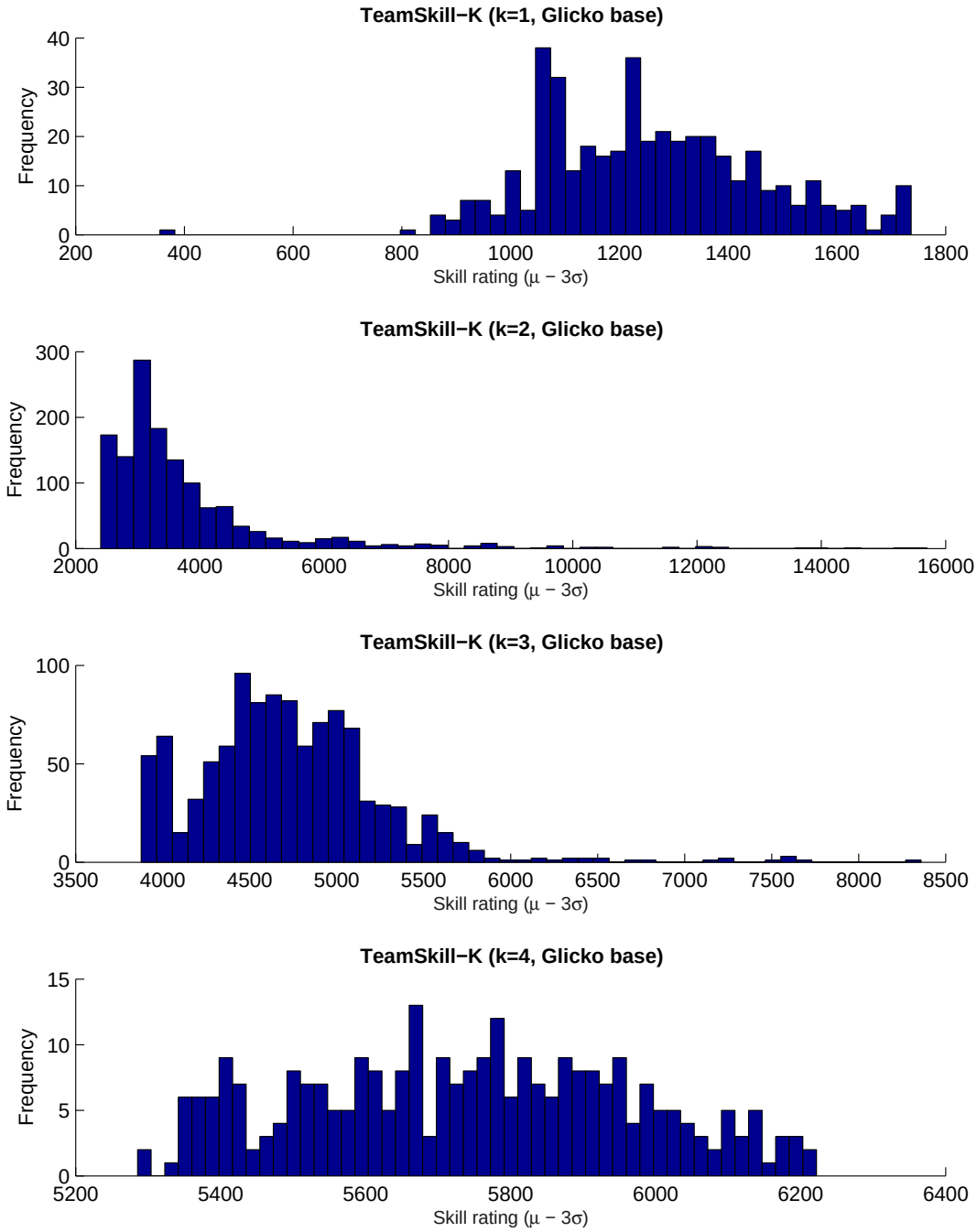


Figure E.2: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with Glicko ($\mu_0 = 1500k, \sigma_0^2 = 100^2 k$) as the base learner.

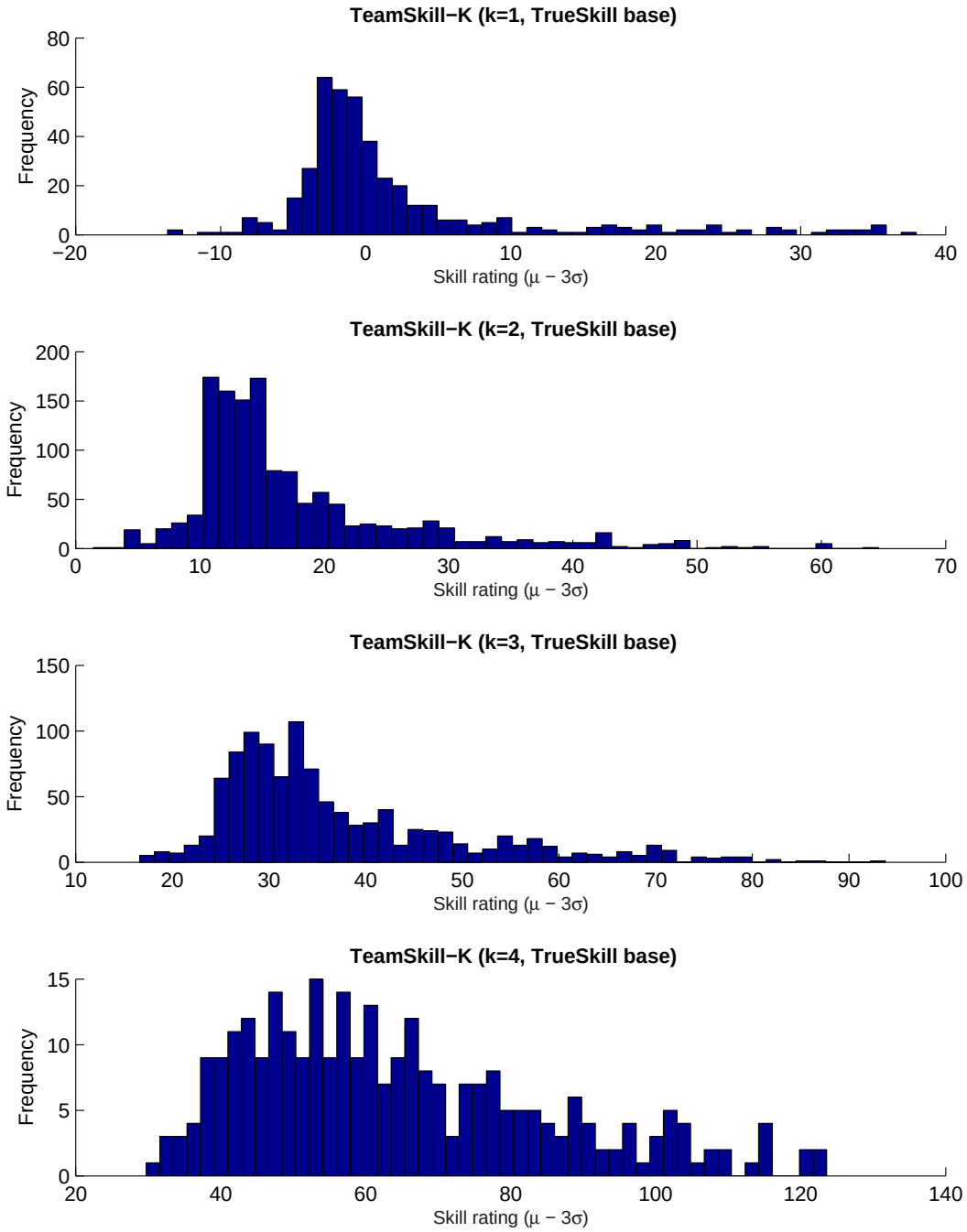


Figure E.3: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with TrueSkill ($\mu_0 = 25k$, $\sigma_0^2 = (\mu_0/3)^2 k$) as the base learner.

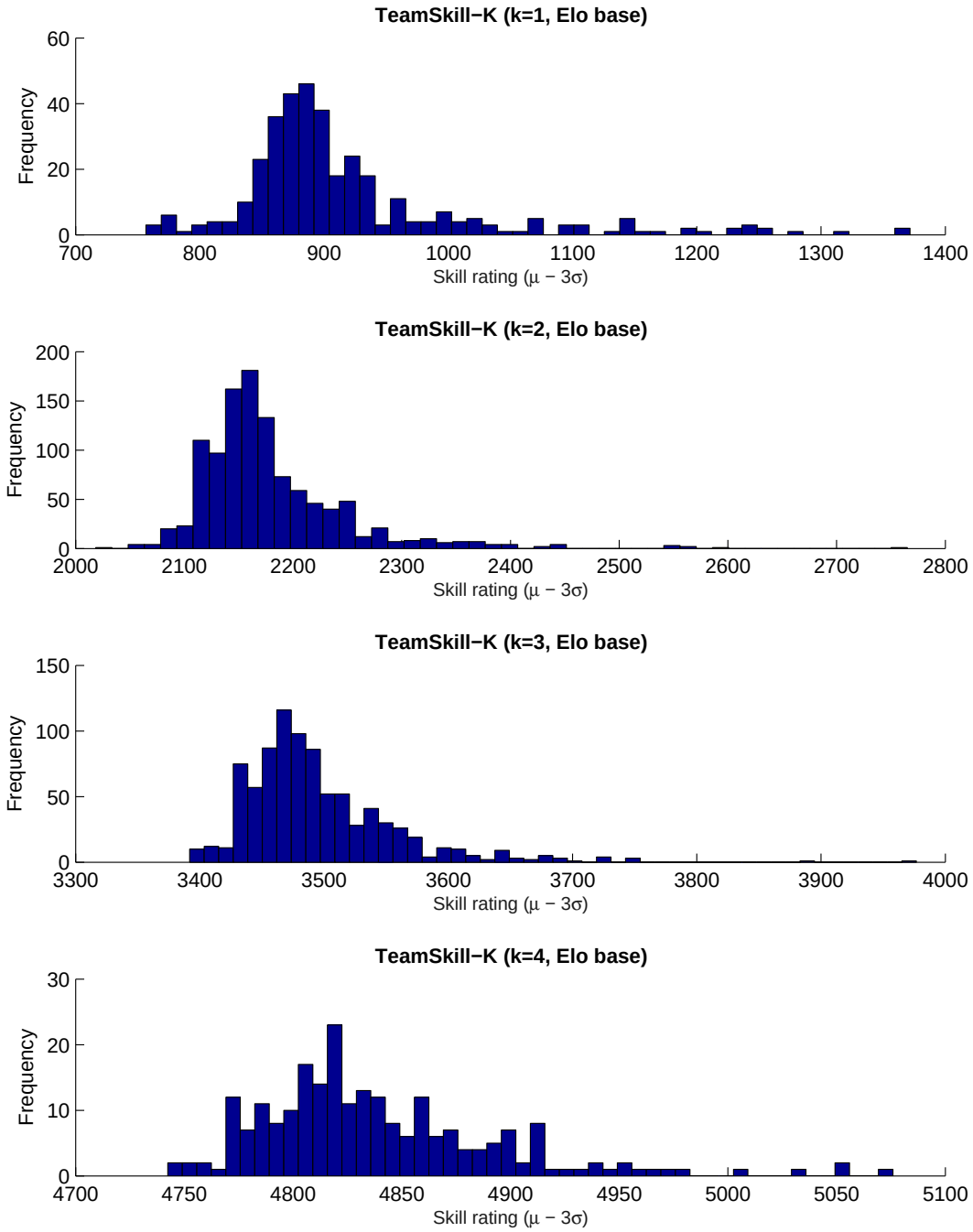


Figure E.4: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with Elo ($\beta = 193.4364$, $\mu_0 = 1500k$, $\sigma_0^2 = \beta^2 k$) as the base learner.

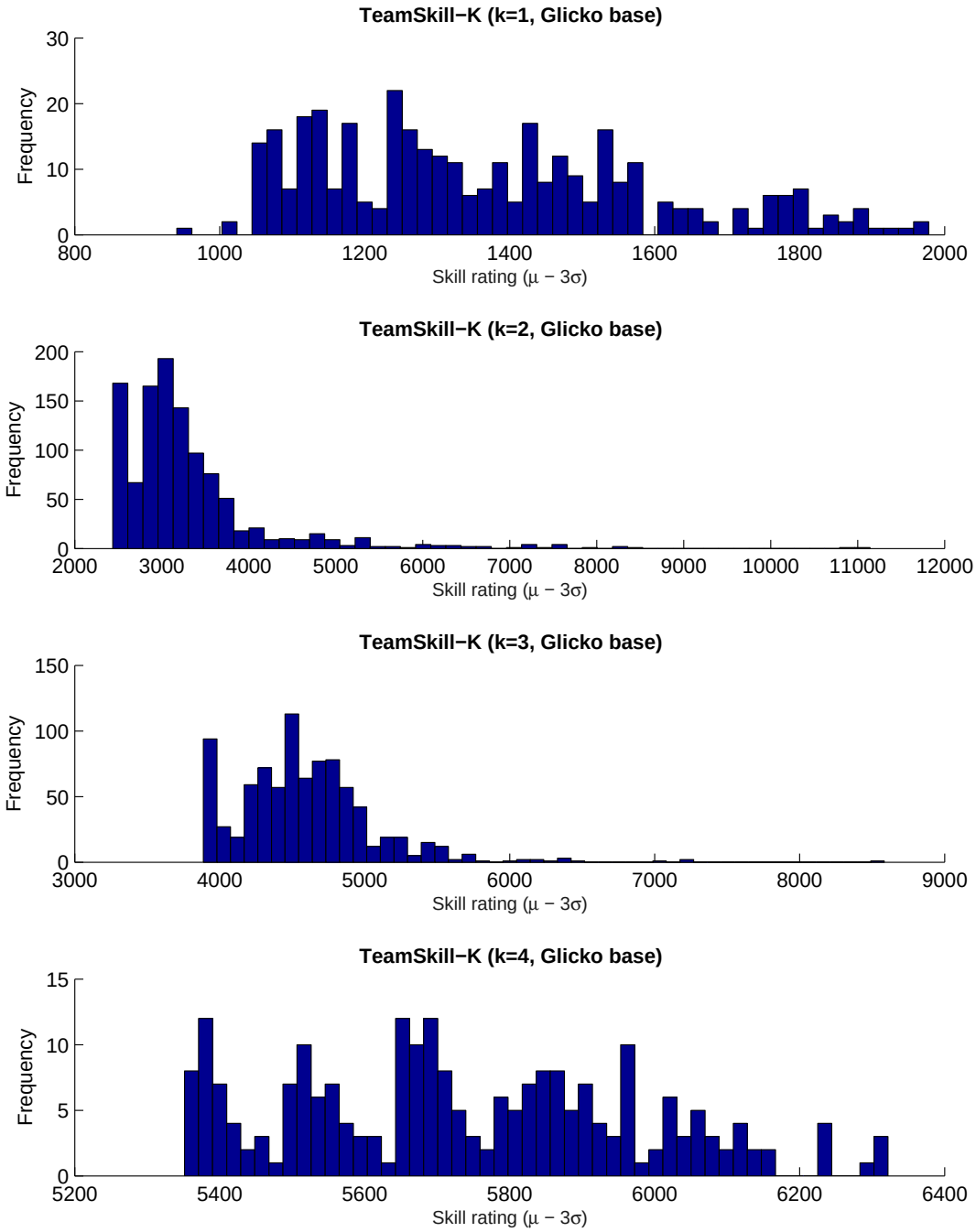


Figure E.5: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with Glicko ($\mu_0 = 1500k, \sigma_0^2 = 100^2 k$) as the base learner.

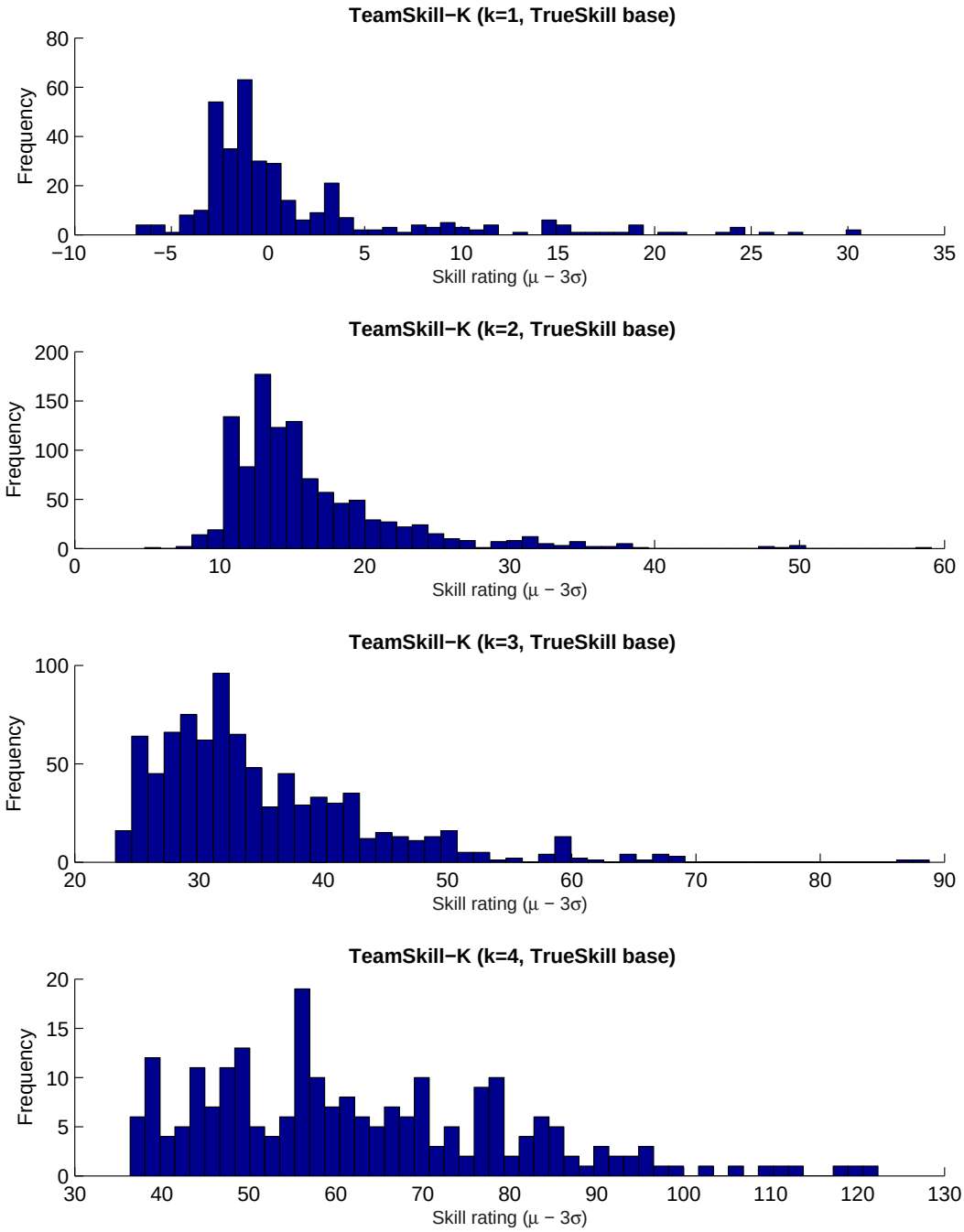


Figure E.6: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with TrueSkill ($\mu_0 = 25k$, $\sigma_0^2 = (\mu_0/3)^2 k$) as the base learner.

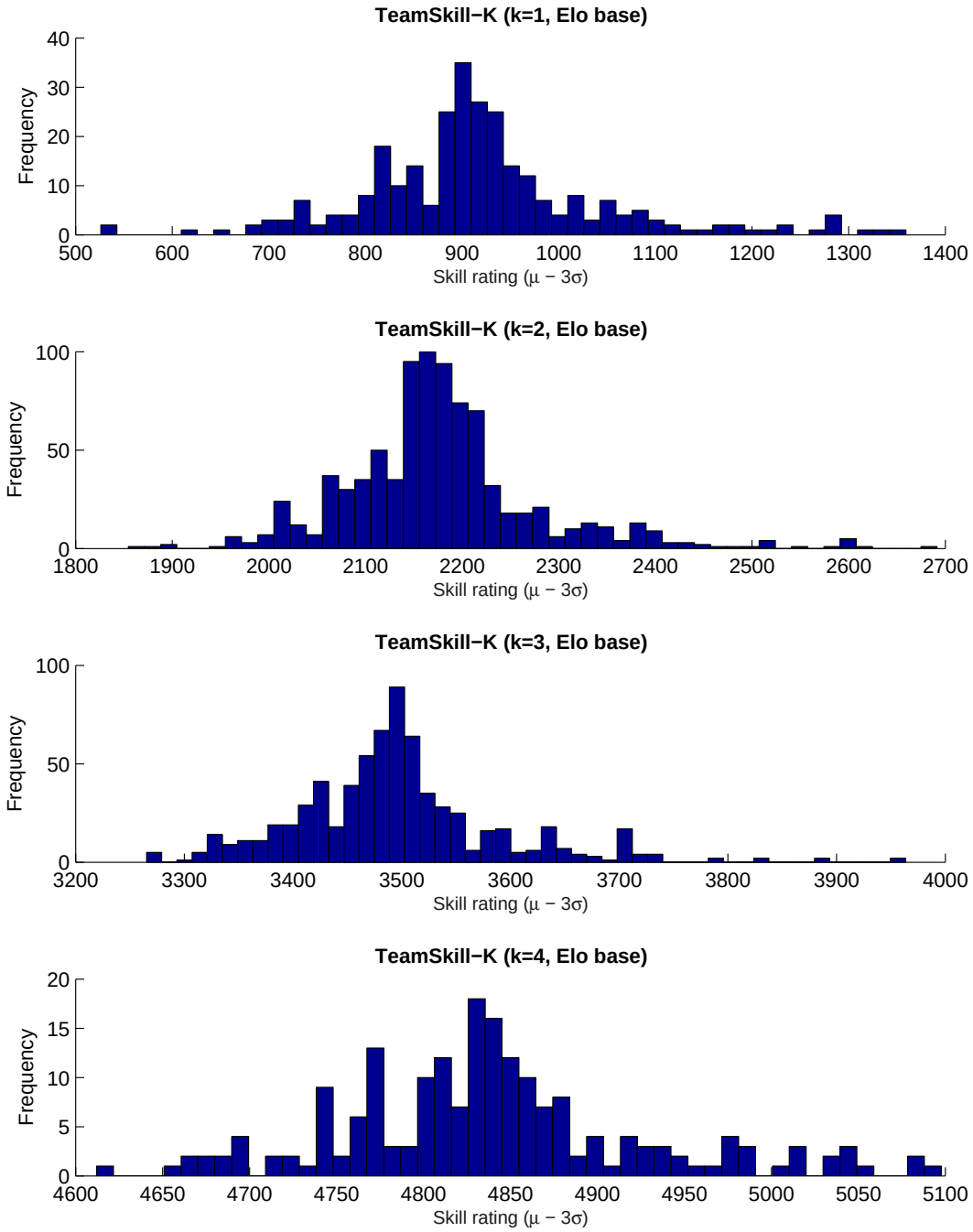


Figure E.7: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with Elo ($\beta = 193.4364$, $\mu_0 = 1500k$, $\sigma_0^2 = \beta^2 k$) as the base learner.

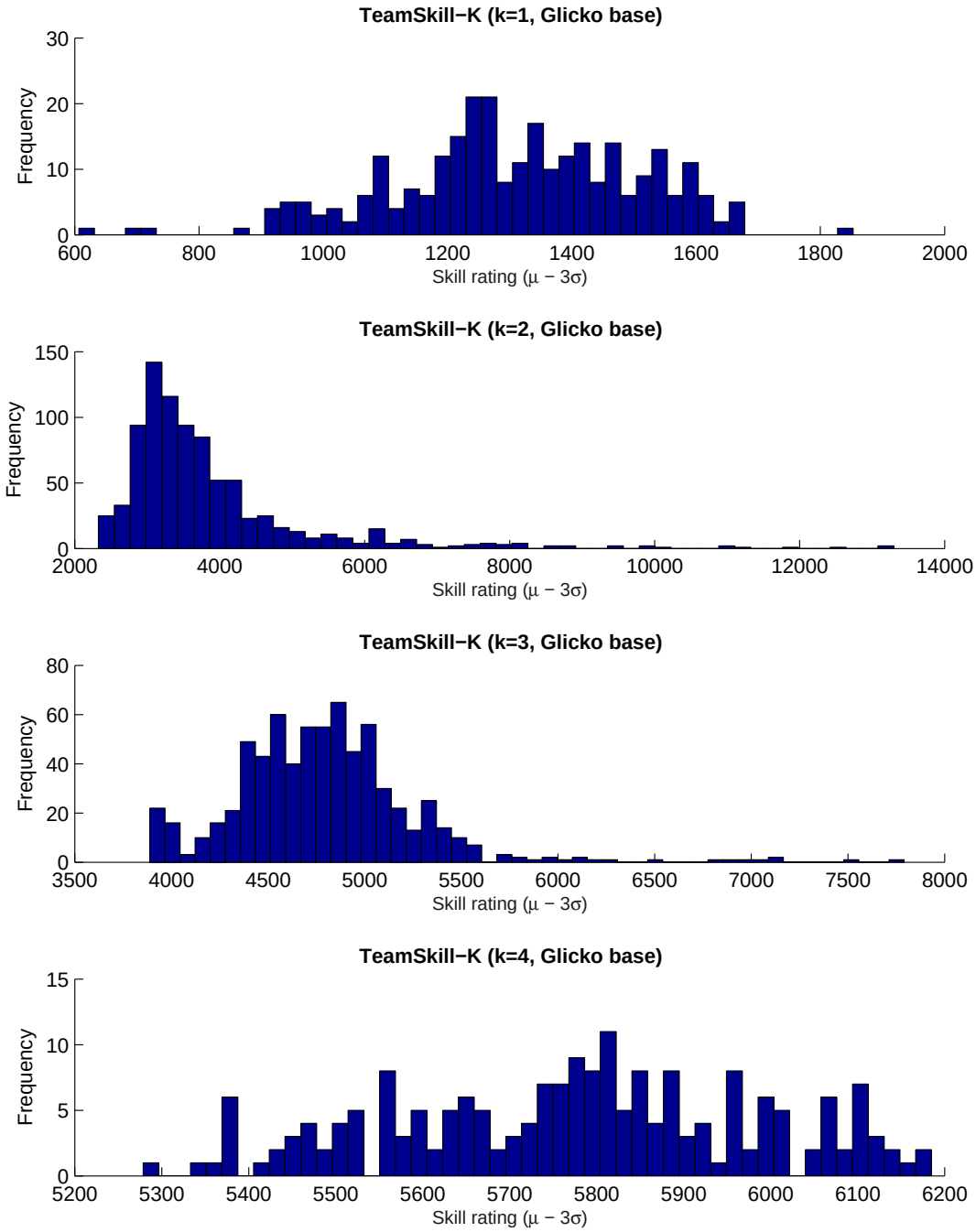


Figure E.8: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with Glicko ($\mu_0 = 1500k, \sigma_0^2 = 100^2 k$) as the base learner.

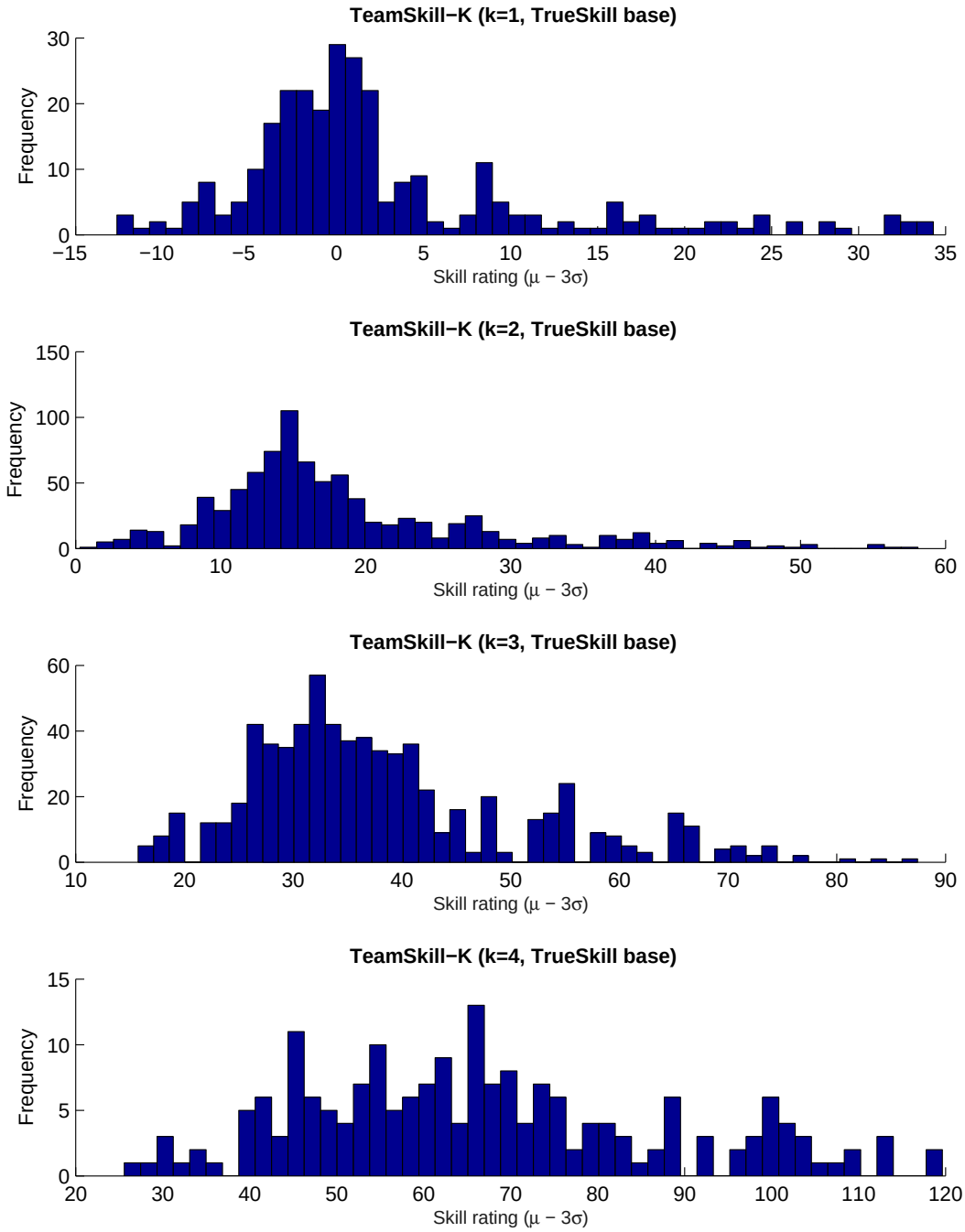


Figure E.9: Histograms of final skill ratings for each subgroup size k for TeamSkill-K with TrueSkill ($\mu_0 = 25k$, $\sigma_0^2 = (\mu_0/3)^2 k$) as the base learner.