Structured Models for Competitive Team Sports

Zhaozhi Qian

Advised by Dr. Franz J. Király

September 2016

This report is submitted as part requirement for the MSc Degree in CSML at University College London. It is substantially the result of my own work except where explicitly indicated in the text. The report may be freely copied and distributed provided the source is explicitly acknowledged. The code used in this project is available at https://github.com/kiraly-group/zhaozhi-qian.msc.

Abstract

Modeling the performance in competitive team sports such as soccer, football or rugby has remained a challenge, partly due to the large number of influential factors which are unobserved. Though predictions which are better than a random guess are possible, the exact nature of a good predictive models, or quantifiers of team performance remain unclear. With more and more data having become available in recent years, selection and empirical assessment of such models has become a possibility. This project will focus on investigating and validating the goodness of popular statistical and machine learning models for the prediction problem of competitive team sports, as well as developing new ones.

Contents

1 Introduction

1.1 Modeling and predicting competitive team sports

Competitive team sports refers to any sport that involves two teams competing against each other to achieve higher scores. Competitive team sports includes some of the most popular and most watched games such as football, basketball and rugby. The sports is played both in domestic professional leagues such as the National Basketball Association, and international competitions such as the FIFA World Cup. For football alone, there are over one hundred fully professional leagues in 71 countries globally. It is estimated that the Premier League, the top football league in the United Kingdom, attracted 4.7 billion television audience last season (League, 2016). Around the world, betting on the outcome of competitive team sports has a long tradition.

The outcome of a match is determined by a large number of factors. Just to name a few, they might involve the competitive strength of each individual player in both teams, the smoothness of collaboration between players, and the team's strategy of playing. Moreover, the composition of the team changes over the years because players leave or join the team. The team composition may also change within the tournament season or even during a match because of injuries or penalties. Keeping track of all these factors is usually unrealistic, and in general we need to assume that unpredictable statistical "noise" is involved in the process. Phenomena which can not be specified deterministically are in fact very common in nature. Earthquakes and volcano eruptions are good examples. Statistics and probability theory provide ways to make inference under randomness. Therefore, modeling and predicting the results of competitive team sports naturally falls into the area of Statistics and machine learning.

Unlike individual sports such as running and swimming where athletics compete against the nature, competitive team sports involves complex interactions between two opposing teams. Therefore, the performance of a team cannot be measured directly using a simple metric. Instead, the strength is measured relative to other teams. This further complicates the prediction for competitive team sports.

Research of modeling competitive sports has a long history. In early days, the research was often closely related to sports betting (Griffith, 1949; Isaacs, 1953). Élő (1978) proposed an influential model that quantifies a Chess player's strength by a numerical rating. The model is also able to predict the chance that a player wins or loses to the opponent. In fact, the Élő model can be applied to any sport that involves two competing teams or players. There has been research into modeling individual sports as well. Recent studies achieved satisfactory performance in predicting individual sports such as athletic running (Blythe and Király, 2015). However, as illustrated later in this thesis, predicting competitive team sports remains to be a difficult problem.

The study of competitive team sports has two main focuses. The first aspect is to design models

that make better prediction for future matches. The second aspect is to understand the key factors that influence the match outcome through retrospective analysis (Pollard, 1986; Rue and Salvesen, 2000). These two aspects are intrinsically connected, and they are the two facets of a single problem. For one thing, researchers can only scientifically defend the proposed influential factors by falsifiable experiments such as predictions on future matches. If the predictive performance does not increase when information about such factors are made available, we should conclude that these factors are actually irrelevant. For another thing, the statistical models are highly likely to make better prediction when the influential factors (also known as "features") become available. This is because the new factors can potentially explain unmodeled random effects (noise). In light of this, the main problem considered in this thesis is the *prediction* problem.

The precondition of all statistical analysis is the availability of data. However, there rarely exists publicly available data set that contains in-match events. Even if such information is available, the data set often only contains matches in one or two recent tournament seasons Fleig (2012). The scarcity of data places a hurdle to the academic research in competitive team sports.

1.2 Overview of the thesis

This thesis is organized in the following parts. Section 2 presents a review of the literature related to the prediction problem of competitive team sports. Section 3 provides extensions of existing models and develops new ones. Section 4 describes the settings and findings of the empirical experiments. Section 4.1 validates new models with synthetic data, and section 4.2 compares the performance of both existing and new models on a real data set. Finally, concluding thoughts and future directions for research are given in section 5.

2 Background and related work

In this section, I will first define the prediction problem mathematically and discuss about the choice of evaluation metric. Next, I will show the implications of sequential data for modeling competitive team sports in section 2.3. Section 2.4 categorizes the existing models into three classes based on their characteristics. The following three sections review each class in more details. Finally, section 2.8 summarizes the evaluation results reported in previous studies.

2.1 Probabilistic supervised prediction

In section 1.1, we argued that the outcome of competitive team sports is determined by a large number of factors. In practice, these factors are often intractable either due to the lack of data

or the complex and dynamic interactions involved in the sports. Under such condition, the outcome of competitive team sports would still display a certain level of randomness even if we have modeled all available information correctly. Therefore, a very desirable feature of our predictive model is the ability to produce a *probabilistic* estimate for all possible match outcomes rather than *deterministically* choose one of them.

The data generation process can be captured by a joint random variable (X_{ij}, Y_{ij}). The variable Y_{ij} models the outcome of the match, and the feature variable X_{ij} models all relevant information including the time of the match. The subscripts indicate that the match is between team i and team j. We will suppose there are N teams in total, $i, j = 1 : N$.

Till now, we have not explicitly defined the meaning of "match outcome". For certain sports such as professional basketball, draw is not possible or very rare. In this case, the *binary* outcome $Y_{ij} \in \{\text{win}, \text{lose}\}$ can summarize the categorical outcome of the match. For other sports, *ternary* outcome $Y_{ij} \in \{\text{win}, \text{lose}, \text{draw}\}$ is more appropriate. The match outcome can also be defined as the final *scores*, $Y_{ij} \in \mathbb{N}^2$. This definition gives a more detailed summary of the match outcome. Also note that the binary and ternary outcomes can be easily derived from the scores.

The feature variable X_{ij} is typically a multivariate random variable $X_{ij} \in \mathbb{R}^k$. Ideally, X_{ij} should contain all available information in the past that helps to make prediction. Importantly, X_{ij} should contain the information about time of the match as the match outcomes are likely to have a temporal dependence. In competitive team sports, X_{ij} will also be likely to contain a component specific to team i and a similar component specific to team j. For example, X_{ij} can include the identity of the two teams. It can also contain a component for the pair of teams to reflect the interaction effect. The variable X_{ij} is also called "explanatory variable" or "covariates" in different literature.

The observations are independent samples of the corresponding random variable (X_{ij}, Y_{ij}). Hence, the observed data set can be represented as

$$\mathscr{D} = \{(x_{ij}, y_{ij})\}$$

where the lower case letters represent random samples.

The model will make predictions on a future data set where only feature x_{ij}'s are observed. Using a similar notation, we represent the future data as $\mathscr{T}$, which is a collection of observed features, and unobserved random outcomes. It is worth noticing that the target outcomes Y_{ij} in future data set $\mathscr{T}$ are random variables instead of realized samples as we have not yet observed future matches.

$$\mathscr{T} = \{(x_{ij}, Y_{ij})\}$$

The model is trained on the observed data $\mathscr{D}$, and will be used on future data $\mathscr{T}$.

The objective for *probabilistic* prediction problem is formulated in **P1**. Essentially, we would

like to find a mapping f that is close to the true distribution of the target variable. The closeness is defined by the expected loss over future data $\mathscr{T}$.

P1: Given training data $\mathscr{D}$, find a mapping $f \in \mathbb{H}$ such that

$$f = \arg\min \left\{ E_{Y \sim \mathscr{D}} \left[L\left(f(x_{ij}), y_{ij} \right) \right] \right\}$$

where the variable y_{ij} can be defined in different ways depending on the task, and function $L(\cdot)$ is the loss function. The set $\mathbb{H}$ is the hypothesis space; it contains all f that the model can capture. Different models generally have different hypothesis spaces. Therefore, depending on the model, the prediction function f may take different parametric form and it can also be nonparametric.

2.2 The evaluation metric

In order to evaluate different models, we need a criterion to measure the goodness of probabilistic predictions. The most common error metric used in supervised classification problems is the prediction accuracy. However, the accuracy is often insensitive to probabilistic predictions. For example, on a certain test case model A predicts win with 60% confidence while model B predicts win with 95% confidence. If the actual outcome is not win, both models are wrong. In terms of prediction accuracy, they are *equally* wrong because both of them made one mistake. Intuitively speaking, we would expect that model B performs worse since it gives a wrong prediction with high confidence. Hence, using accuracy or misclassification rate alone is insufficient for model comparison.

There exists two common criteria that take into account the probabilistic nature of predictions. The first one is the Brier score (1) and the second is the out-of-sample log-likelihood (2). Both metrics are computed over a separate testing data set which composed of Δt test cases.

$$\hat{\mathrm{BS}} = \frac{1}{\Delta t} \sum_{t=T+1}^{T+\Delta t} \sum_{k=1}^{K} \left(\hat{p}_k(t) - \delta(y_{ij}(t) = k) \right)^2 \tag{1}$$

$$\hat{L} = \frac{1}{\Delta t} \sum_{t=T+1}^{T+\Delta t} \sum_{k=1}^{K} \delta(y_{ij}(t) = k) \log(\hat{p}_k(t)) \tag{2}$$

where $\hat{p}_k(t)$ is the predicted probability given by $f(x_{ij}(t))$ that the actual outcome $y_{ij}(t) = k$, and $\delta(\cdot)$ is the indicator function.

The Brier score is a direct application of the mean squared error used in regression problems. In fact, it is an estimation of the expected mean squared error of predicted confidence scores (3).

In certain cases, Brier score is hard to interpret and it might be unintuitive (Jewson, 2004).

$$\mathrm{BS} = E_{Y\sim\mathscr{T}}\left[\left(\hat{p}_k(t) - \delta(y_{ij}(t) = k)\right)^2\right] \tag{3}$$

On the contrary, the out-of-sample log-likelihood is closely related to the log-likelihood function used in many statistical procedures. The only difference is that the out-of-sample log-likelihood is calculated from the testing data (hence the name out-of-sample). Calculating error metric from testing data is necessary to prevent optimism and over-fitting. The metric is an estimation of the expected log-likelihood over future data (4). Given its interpretability, we would prefer to use the out-of-sample log-likelihood as evaluation metric for prediction models of competitive team sports.

$$L = E_{Y\sim\mathscr{T}}\left[\sum_{k=1}^{K}\delta(y_{ij}(t) = k)\log(\hat{p}_k(t))\right] \tag{4}$$

2.3 Working with sequential data

In reality, the match outcome data is gathered through time. The data set might display a temporal structure. The sequential nature of the data poses another challenge to predicting competitive team sports. This has three consequences to the scientific study of sports prediction.

The most obvious consequence is that the model will need to take into account the temporal structure in some way. A common approach for statistical models is to assume a temporal structure in the latent variables that determine a team's strength. A different and somewhat ad-hoc approach proposed by Dixon and Coles (1997) is to assign lower weights to earlier observations and estimate parameter by maximizing the weighted log-likelihood function. For machine learning models, the temporal structure is often encoded with handcrafted features.

Another consequence concerns about the way to update the model. Since information about match outcomes is revealed through time, the model needs to be updated correspondingly. The online methods update model parameters after each new match outcome is revealed. However, the batch methods need to be retrained after new information arrives. Ideally, the model is retrained with all historical data each time a new observation is made, but this approach is often unpractical. A common alternative is to re-train the model after a certain amount of time (a week, a month, etc). There is no general consensus about how frequent the retraining should be performed.

Last and equally importantly, the sequential nature of the data has significant implications for model evaluation and model comparison. We can no longer assume that the evaluation metrics computed over the testing data are independent. This is because the outcomes of two consecutive matches are likely to be dependent. Hence, a common assumption made in many validation methods is violated. In particular, the K-fold cross validation method (Stone, 1974) assumes that data

within each fold are independent. It will underestimate the variance of the error metric. Moreover, the validation method will need to accommodate the fact that the model may be updated online during testing. In the literature, the validation method for data with temporal structure is largely an unexplored area. Developing a reasonable validation method is crucial for scientifically assessing the models. A plausible validation method is introduced in section 4.2.2 in detail.

We should keep these three aspects in mind when studying the competitive team sports.

2.4 A brief summary of previous studies

The Élő model is one of the earliest and most well-known attempts to model competitive sports (Élő, 1978). The model only uses information about the historical match outcomes. The Élő model associates each team with a parameter, the Élő rating. The rating reflects a team's competitive skills: the team with higher rating is stronger. More importantly, the Élő model gives a probabilistic prediction for the *binary* match outcome based on the ratings of two teams. The parametric form of the Élő model allows the ratings to be updated efficiently after each observed match. The online update further enables the model to capture the temporal dependence between matches in a juristic way.

Many researchers have adopted a similar approach: their models parametrize the distribution of match outcomes explicitly. Maher (1982) studied the problem of predicting the final scores. The proposed model assumes that the two scores are independent Poisson random variables whose means depend on the competing teams. Dixon and Coles (1997) improved Maher's model by introducing a correlation effect between the two final scores.

Since late 1990s, researchers have explored the Bayesian approach to model competitive team sports. These studies have been focused on predicting the final scores. Similar to previous studies, the scores are assumed to follow a parametric distribution. The Bayesian approach opens up many ways to model the temporal structure. Different latent variable models and hierarchical models were proposed to capture the temporal dependence (Glickman and Stern, 1998; Rue and Salvesen, 2000; Crowder et al., 2002). The Bayesian parametric models also allow researchers to inject expert knowledge through the prior distribution. The prediction function is naturally given by the posterior distribution of the scores, which can be updated as more observations become available.

Parametric models are often simple, and the parameters are often interpretable as indicators of team strength. However, most parametric models only take historical match outcomes as features (see Constantinou et al. (2012) for an exception). This limitation becomes more obvious when researchers gain access to more varied information. Some researchers have therefore started to explore the use of machine learning predictors in the context of sports prediction. Unlike parametric models, the machine learning predictors were developed as general supervised learning techniques

rather than models specific to the domain of sports modeling. Their performance thus depends crucially on the "features" available to them. Moreover, most machine learning models investigated by researchers so far are trained in-batch, and they need to be re-trained periodically to incorporate new observations.

More detailed literature overview on the Élő model will be given in 2.5. Section 2.6 reviews other parametric models for predicting final scores. After that, section 2.7 reviews the machine learning predictors and feature engineering for sports prediction.

2.5 The Élő model

2.5.1 The probabilistic interpretation of the Élő model

The simplified Élő model We will first introduce the simplified version of the Élő model (Élő, 1978). The probabilistic formulation of this model first appears in Glickman (1995). The simplified Élő model assumes that there is a latent random variable Z_i associating with team i. The latent variables are statistically independent and they follow a specific generalized extreme value (GEV) distribution:

$$Z_i \sim \text{GEV}(\theta_i, 1, 0)$$

where the mean parameter θ_i varies across teams, and the other two parameters are fixed at one and zero. The parameter θ's are called the Élő ratings. The density function of $\text{GEV}(\mu, 1, 0)$, $\mu \in \mathbb{R}$ is

$$f(z) = \exp(-(x-\mu)) \cdot \exp(-\exp(-(x-\mu)))$$

The model further assumes that team i wins over team j in a match if and only if a random sample (z_i, z_j) from the associated latent variables satisfies $z_i > z_j$. In fact, for any i and j, the difference $(Z_i - Z_j)$ follows a logistic distribution with mean $\theta_1 - \theta_2$ and scale parameter 1 (Resnick, 2013). Hence, the winning probability for team i is given by equation 5 [1]. From this equation, we can see that the winning probability only depends on the difference in Élő ratings. This means that the unknown parameters in the model are not uniquely identifiable (we can add or subtract a constant to all parameters, and the model will give the same probability for all match outcomes). Also note that the draw probability $P(Z_i = Z_j)$ is always zero because Z_i and Z_j are continuous random variables.

$$p_{ij} = P(Z_i - Z_j > 0) = \frac{1}{1 + e^{-(\theta_i - \theta_j)}} = \sigma(\theta_i - \theta_j) \tag{5}$$

[1] In practice, it is more common to use $p_{ij} = \frac{1}{1+10^{-(\theta_i-\theta_j)/400}}$. The two formulas are equivalent if ratings are properly scaled

There exists a different interpretation of simplified Élő model. If the task is to predict match outcomes, the latent variables are no longer necessary. The *binary* match outcome between team i and team j can be directly modeled as a Bernoulli random variable with success probability p_{ij} given in 5. The set of unknown parameters θ is the same, and the model has the same expressiveness. The new formulation is more succinct and it is the foundation of a new family of models: the structured log-odds model (section 3.1).

The Élő model In many competitive team sports, the home team is likely to have an advantage. The standard version of Élő model introduces a hyper-parameter h to model the home effect. If team i is the home team, the winning probability is modified to be (6). Section 3 shows that this modification has a connection with the logistic regression.

$$p_{ij} = \sigma(\theta_i - \theta_j + h) \tag{6}$$

Assuming that the outcome of matches are independent and draws are not possible, the log-likelihood function of N match outcomes is given by

$$\ell(\theta) = \sum_{n=1}^{N} \left[y_{ij}(n) \log(p_{ij}) + \big(1 - y_{ij}(n)\big) \log(1 - p_{ij}) \right] \tag{7}$$

where $y_{ij} = 1$ if team i wins, and $y_{ij} = 0$ otherwise. The n in the brackets represents the index of match in the data set.

2.5.2 The online update of the Élő model

The update rule for the Élő rating for team i after observing a match between team i and team j is given by:

$$\theta_i \leftarrow \theta_i + K \left[S_{ij} - p_{ij} \right] \tag{8}$$

where θ_i is the Élő rating, S_{ij} represents the outcome of the match: $S_{ij} = 1$ if team i wins, $S_{ij} = 0$ if team i loses and $S_{ij} = 0.5$ if there is a draw. The p_{ij} is the predicted probability given by the model before the match. K is a hyper-parameter called the "K factor". A similar update is performed on the rating of team j as well.

As explained in (Glickman, 1995), the update rule (8) makes sense intuitively because the term $(S_{ij} - p_{ij})$ can be thought of as the discrepancy between what is expected (p_{ij}) and what is observed (S_{ij}). The update will be bigger if the current parameter setting produces a large discrepancy. However, the theoretical justification has not been articulated in the literature. If fact,

Élő himself commented that “the logic of the equation is evident without algebraic demonstration” (Élő, 1978).

We will show the online update of the Élő model is a Stochastic Gradient Ascent procedure on the log-likelihood function in section 3.3.

2.5.3 Limitations of the Élő model and existing remedies

Modeling draws The Élő model does not model the possibility of a draw. This might be reasonable in some Chess tournaments where players play on until draws are resolved. However, in many competitive sports a significant number of matches end up as a draw. For example, in the English Premier League about twenty percent of the matches end up as a draw. Modeling the possibility of draw outcome is therefore very relevant. A method to model draws is proposed in section 3.2.1.

Using final scores in the model The Élő model only takes into account the binary outcome of the match. In sports such as football, the final scores for both teams may contain more information. Generalizations exist to tackle this problem. One approach is adopted by the official FIFA Women’s football ranking (FIFA, 2016), where the actual outcome of the match is replaced by the "Actual Match Percentage", a quantity that depends on the final scores. FiveThirtyEight, an online media, proposed another approach (Silver, 2014). It introduces the “Margin of Victory Multiplier” in the rating system to adjust the K factor for different final scores.

In a survey paper, Lasek et al. (2013) showed empirical evidence that rating methods that take into account the final scores often outperform those do not. However, it is worth noticing that the existing methods often rely on heuristics and their mathematical justifications are often unpublished or unknown. We will propose a principled way to incorporate final scores in section 3.2.3.

Using additional features The Élő model only takes into account very limited information. Apart from previous match outcomes, the only feature it uses is the identity of home and away teams. There are many other potentially useful features. For example, whether the team is recently promoted from a lower-division league, or whether a key player is absent from the match. These features may help make better prediction if they are properly modeled. In section 3.2.2, we will introduce the relationship between the Élő model and logistic regression. Then we will be able to extend the Élő model to use more features.

2.6 Domain-specific parametric models

2.6.1 Bivariate Poisson regression and extensions

Maher (1982) proposed to model the final scores as independent Poisson random variables. If team i is playing at home field against team j, then the final scores S_i and S_j follows

$$\begin{aligned} S_i &\sim \text{Poisson}(\alpha_i \beta_j h) \\ S_j &\sim \text{Poisson}(\alpha_j \beta_i) \end{aligned}$$

where α_i and α_j measure the 'attack' rates, and β_i and β_j measure the 'defense' rates of the teams. The parameter h is an adjustment term for home advantage. The model further assumes that all historical match outcomes are independent. The parameters are estimated from maximizing the log-likelihood function of all historical data. Empirical evidence suggests that the Poisson distribution fits the data well. Moreover, Poisson distribution can be derived as the expected number of events during a fixed time period at a constant risk. This interpretation fits into the framework of competitive team sports.

Dixon and Coles (1997) proposed two modifications to Maher's model. First, the final scores S_i and S_j are allowed to be correlated when they are both less than two. The model employs a free parameter ρ to capture this effect. The joint probability function of S_i, S_j is given by the bivariate Poisson distribution 9:

$$P(S_i = s_i, S_j = s_j) = \tau_{\lambda,\mu}(s_i, s_j) \frac{\lambda^{s_i} \exp(-\lambda)}{s_i!} \cdot \frac{\lambda^{s_j} \exp(-\mu)}{s_j!} \tag{9}$$

where

$$\begin{aligned} \lambda &= \alpha_i \beta_j h \\ \mu &= \alpha_j \beta_i \end{aligned}$$

and

$$\tau_{\lambda,\mu}(s_i, s_j) = \begin{cases} 1 - \lambda\mu\rho & if\, s_i = s_j = 0, \\ 1 + \lambda\rho & if\, s_i = 0, s_j = 1, \\ 1 + \mu\rho & if\, s_i = 1, s_j = 0, \\ 1 - \rho & if\, s_i = s_j = 1, \\ 1 & otherwise. \end{cases}$$

The function $\tau_{\lambda,\mu}$ adjusts the probability function so that drawing becomes less likely when both scores are low. The second modification is that the Dixon-Coles model no longer assumes match

outcomes are independent through time. The modified log-likelihood function of all historical data is represented as a weighted sum of log-likelihood of individual matches illustrated in equation 10, where t represents the time index. The weights are heuristically chosen to decay exponentially through time in order to emphasize more recent matches.

$$\ell = \sum_{t=1}^{T} \exp(-\xi t) \log \left[P(S_i(t) = s_i(t), S_j(t) = s_j(t)) \right] \tag{10}$$

The parameter estimation procedure is the same as Maher's model. Estimates are obtained from batch optimization of modified log-likelihood.

Karlis and Ntzoufras (2003) explored several other possible parametrization of the bivariate Poisson distribution including those proposed in Kocherlakota and Kocherlakota (1992), and Johnson et al. (1997). The authors performed a model comparison between Maher's independent Poisson model and various bivariate Poisson models based on AIC and BIC. However, the comparison did not include the Dixon-Cole model. Goddard (2005) performed a more comprehensive model comparison based on their forecasting performance.

2.6.2 Bayesian latent variable models

Rue and Salvesen (2000) proposed a Bayesian parametric model based on the bivariate Poisson model. In addition to the paradigm change, there are three major modifications on the parameterization. First of all, the distribution for scores are truncated: scores greater than four are treated as the same category. The authors argued that the truncation reduces the extreme case where one team scores many goals. Secondly, the final scores S_i and S_j are assumed to be drawn from a mixture model:

$$P(S_i = s_i, S_j = s_j) = (1 - \varepsilon) P_{DC} + \varepsilon P_{Avg}$$

The component P_{DC} is the truncated version of the Dixon-Coles model, and the component P_{Avg} is a truncated bivariate Poisson distribution (9) with μ and λ equal to the average value across all teams. Thus, the mixture model encourages a reversion to the mean. Lastly, the attack parameters α and defense parameters β for each team changes over time following a Brownian motion. The temporal dependence between match outcomes are reflected by the change in parameters. This model does not have an analytical posterior for parameters. The Bayesian inference procedure is carried out via Markov Chain Monte Carlo method.

Crowder et al. (2002) proposed another Bayesian formulation of the bivariate Poisson model based on the Dixon-Coles model. The parametric form remains unchanged, but the attack parameters α_i's and defense parameter $\beta_i' s$ changes over time following an AR(1) process. Again, the model does not have an analytical posterior. The authors proposed a fast variational inference

procedure to conduct the inference.

Baio and Blangiardo (2010) proposed a further extension to the bivariate Poisson model proposed in Karlis and Ntzoufras (2003). The authors noted that the correlation between final scores are parametrized explicitly in previous models, which seems unnecessary in the Bayesian setting. In their proposed model, both scores are *conditionally* independent given an unobserved latent variable. This hierarchical structure naturally encodes the *marginal* dependence between the scores.

2.7 Feature-based machine learning predictors

In recent publications, researchers reported that machine learning models achieved good prediction results for the outcomes of competitive team sports. The advantage of machine learning predictors is that researchers can easily add new features to the model. When using the machine learning algorithms, the researchers usually formulate the prediction problem as a three-class classification problem, where the outcome of a match falls into three distinct classes: home team win, draw, and home team lose. A less-explored approach is to treat the match outcomes as ordinal variables: home team win $\succ$ draw $\succ$ away team win. This approach assumes that draw is a middle outcome, and it might be a reasonable assumption in many cases.

Liu and Lai (2010) applied logistic regression, Support Vector Machines with different kernels, and AdaBoost to predict NCAA football outcomes. For this prediction problem, the researchers hand crafted 210 features.

Hucaljuk and Rakipović (2011) explored more machine learning predictors in the context of sports prediction. The predictors include Naïve Bayes, Bayes networks, LogitBoost, k-nearest neighbors, Random forest, and Artificial neural networks. The models are trained on 20 features derived from previous match outcomes and 10 features designed subjectively by experts (such as team's morale).

Odachowski and Grekow (2012) conducted a similar study. The predictors are commercial implementations of various Decision Tree and ensembled trees algorithms as well as a hand-crafted Bayes Network. The models are trained on a subset of 320 features derived form the time series of betting odds. In fact, this is the only study so far where the predictors have no access to previous match outcomes.

Kampakis and Adamides (2014) explored the possibility of predicting match outcome from Tweets. The authors applied Naïve Bayes, Random forests, logistic regression, and Support Vector Machines to a feature set composed of 12 match outcome features and a number of Tweets features. The Tweets features are derived from unigrams and bigrams of the Tweets.

2.8 Evaluation methods used in previous studies

In all studies mentioned in this section, the authors validated their new model on a real data set and showed that the new model performs better than an existing model. However, complication arises when we would like to aggregate and compare the findings made in different papers. Different studies may employ different validation settings, different evaluation metrics, and different data sets. The study may or may not include a well-chosen benchmark for comparison. In some studies, the variable selection or hyper-parameter tuning procedure is not described explicitly, which raises doubts about the validity of conclusions. Last but equally importantly, some studies do not report the error measure on evaluation metrics (standard deviation or confidence interval). In these studies, we cannot rule out the possibility that the new model is outperforming the baselines just by chance.

In table 1, we summarize the evaluation methods used in previous studies. We've noticed that the size of testing data sets vary considerably across different studies, and most studies do not provide a quantitative assessment on the evaluation metric. We also note that some studies perform the evaluation on the training data. These evaluation results can only show the goodness-of-fit of the model, and they do not provide much insight into the model's predictive performance.

Study	Validation	Tuning	Problem	Metric	Baseline	Error	Train	Test
Lasek et al. (2013)	Online	Yes	Binary	Brier score, Binomial divergence	Yes	Yes	-	979
Maher (1982)	In sample	No	Scores	χ^2 statistic	No	No	5544	-
Dixon and Coles (1997)	No	No	Scores	Non-standard	No	No	-	-
Karlis and Ntzoufras (2003)	In sample	-	Scores	AIC, BIC	No	No	615	-
Goddard (2005)	Custom	-	Scores	oos-log-lik	No	No	6930	4200
Rue and Salvesen (2000)	Custom	-	Scores	oos-log-lik	Yes	No	280	280
Crowder et al. (2002)	Online	-	Tenary	Accuracy	No	No	1680	1680
Baio and Blangiardo (2010)	Hold out	-	Scores	Not reported	No	No	4590	306
Liu and Lai (2010)	Hold out	No	Binary	Accuracy	Yes	No	480	240
Hucaljuk and Rakipović (2011)	Custom	Yes	Binary	Accuracy, F1	Yes	No	96	96
Odachowski and Grekow (2012)	CV-10	No	Tenary	Accuracy	Yes	No	1116	1116
Kampakis and Adamides (2014)	LOO-CV	No	Binary	Accuracy, Cohen's kappa	No	Yes	NA	NA

Table 1: Evaluation methods used in previous studies: the column "Validation" lists the validation settings ("Hold out" uses a hold out test set, "CV-10" refers to the 10-fold cross validation, "LOO-CV" refers to the leave-one-out cross validation, "Online" refers to online prediction, "In sample" means the evaluation metric is computed on training set, "Custom" refers to a customized evaluation method); the column "Tuning" lists whether the tuning method is reported (The Bayesian methods do not need hyper-parameter tuning); "Problem" specifies the prediction problem; the column "Metric" shows the evaluation metric reported; "Baseline" shows whether baselines are reported; "Error" shows whether estimated error of the evaluation metric is reported; "Test" shows the size of testing data; "Train" shows the size of training data.

3 Methods

In this section, we propose a new family of models for the outcome of competitive team sports, the structured log-odds model. We will show that the original Élő model belongs to this family (section 3.1). We then propose several new models with added flexibility (section 3.2) and introduce various training algorithms (section 3.3 and 3.4).

3.1 The structured log-odds model

3.1.1 Motivation and definition of the structured log-odds model

Motivation of structural assumptions As we have discussed in section 2.5.1, the binary match outcome of team i playing against team j at home can be modeled as a Bernoulli random variable with success probability p_{ij}. For the time being, we ignore the temporal dependency and drop the time index t. The optimal prediction rule under this model is to always predict the true probability p_{ij}, which is unfortunately unknown to us. Therefore, the prediction problem can be viewed as an estimation problem where we infer the unknown quantities p_{ij} for all i and j's.

Let us use Φ to denote the probability matrix whose (i, j) entry is p_{ij}. If team i is a strong team, we should expect p_{ij} to be larger for all j's. We can make a similar argument if we know team i is a weak team. This means the entries in matrix Φ are not completely independent from each other; in other words, the matrix has a structure. We may obtain a better estimation ofΦ if we make the correct structural assumption.

Model definition We are now ready to introduce the structured log-odds model for competitive team sports. The model assumes that log-odds matrix L is a structured matrix, $L_{ij} = \log(\frac{p_{ij}}{1-p_{ij}})$ [2]. The binary match outcome Y_{ij} is modeled as a Bernoulli random variable with probability $p_{ij} = \sigma(L_{ij})$, where $\sigma(\cdot)$ is the sigmoid function. In summary, the model can be written explicitly as

$$
\begin{aligned}
Y_{ij} &\sim \text{Bernoulli}(p_{ij}) && (11)\\
p_{ij} &= \sigma(L_{ij}) && (12)\\
\text{Matrix}\, L &= [L_{ij}]\, \text{has a structure} && (13)
\end{aligned}
$$

There are many different structural assumptions for matrix. A common one is the low-rank assumption: the rank of the unknown matrix is less than a specific value k. Typically, k is far less than the size of the matrix. The low-rank assumption essentially reflects our belief that the unknown matrix is determined by only a small number of factors. Many machine learning applications can be viewed as estimation or completion of a low-rank matrix (So and Ye, 2007; Vounou et al., 2010).

An assumption that is specific to match prediction is the antisymmetric assumption. If we assume that all matches are played on neutral fields, we should expect that $p_{ij} = 1 - p_{ji}$, which means the probability for team i to beat team j is the same regardless of where the match is played.

[2]For a technical reason, we would focus on estimating the log-odds matrix instead of the probability matrix. The range of log-odds is $\Re$ whereas the range of probability is [0, 1]. Using log-odds removes the boundary constraints.

In this case, L is an antisymmetric matrix: $L+L^\top = 0_{N\times N}$. However, it is widely believed that in certain sport such as basketball the home team has an advantage. The antisymmetric assumption will not be sensible for these sports. An interesting mathematical fact is that a real antisymmetric matrix always has even rank (Eves, 1980). Moreover, the eigenvalues of a real antisymmetric matrix come in pairs $\pm\lambda$. In general, a real antisymmetric matrix with rank $2k$ can be decomposed into the combination of $2k$ independent factors (14).

$$L = \sum_{i=1}^{k} \left(u_i \cdot v_i^\top - v_i \cdot u_i^\top \right) \tag{14}$$

Another useful structural assumption is that a factor in the matrix decomposition (14) is a vector of ones. As we will see later, the one factor may have different interpretation under different scenarios.

The exact structural assumption of the model will affect the estimation of p_{ij}, but the log-likelihood function has the same form for all structured log-odds models (15). This will help us derive a general framework for training these models.

$$l(\theta) = \sum_{n=1}^{N} \left[Y_{ij}(n) \log(p_{ij}) + \left(1 - Y_{ij}(n)\right) \log(1 - p_{ij}) \right] \tag{15}$$

3.1.2 Connection to existing models

Recalling equation 5, we recognize that the log-odds matrix L in the simplified version of the Élő model is given by $L_{ij} = \log(\frac{P_{ij}}{1-P_{ij}}) = \theta_i - \theta_j$, where θ_i and θ_j are the Élő ratings. Using the rule of matrix multiplication, one can verify that

$$L = \theta \cdot \underline{1}^\top - \underline{1} \cdot \theta^\top$$

where $\underline{1}$ is a vector of ones and θ is the vector of Élő ratings. If not all $\theta_i = \theta_j$, the log-odds matrix will have rank two. It is clear that the simplified Élő model makes all three structural assumptions: the log-odds matrix is a low-rank antisymmetric matrix with a factor of ones. Hence, the simplified Élő model is a special case of the structured log-odds model. The factor of ones indicates that the winning chance depends on the difference in factor θ between two teams.

From equation (6), we can recognize that the standard Élő model decomposes the log-odds matrix as

$$L = \theta \cdot \underline{1}^\top - \underline{1} \cdot \theta^\top + \underline{1} \cdot \underline{1}^\top \cdot h$$

The log-odds matrix is no longer antisymmetric thanks to the term with home advantage parameter h.

We can further increase the expressiveness of the simplified Élő model by relaxing other structural assumptions. Two examples of structured log-odds model are the two-factor Élő model and the rank-four Élő model.

The *two-factor Élő model* assumes that the log-odds matrix L is a general rank-two antisymmetric matrix (16). The team's competitive strength is determined by two interacting factors u, v.

$$L = u \cdot v^\top - v \cdot u^\top \tag{16}$$

The *rank-four Élő model* parametrizes the log-odds matrix L as (17), where $\underline{1}$ is a vector of ones. The team's competitive strength is captured by three factors u, v and θ. If the factors are linearly independent, the log-odds matrix would have rank four.

$$L = u \cdot v^\top - v \cdot u^\top + \theta \cdot \underline{1}^\top - \underline{1} \cdot \theta^\top \tag{17}$$

The structured log-odds model is also closely related to logistic regression, a supervised learning model. Similar to the structured log-odds model, the target outcomes are binary variables following Bernoulli distribution. Both models utilize the log-odds for prediction. Logistic regression models the log-odds as a linear combination of features. Given feature x, the log-odds is parametrized as $L = \beta^\top x$, where β is the vector of free parameters.

It is also straightforward to see that the simplified Élő model (2.5.1) is equivalent to logistic regression with a specific indicator feature. Consider a match between team i and team j. If the i^{th} element in x_{ij} is 1, the j^{th} element is -1, and all other elements are 0, we get the log-odds given by the Élő model: $L_{ij} = \beta_i - \beta_j$. In this case, the parameter vector β corresponds to the Élő ratings.

Section 3.2.2 shows that we are able to combine structured log-odds model and logistic regression to further increase the expressiveness.

3.2 Extensions of structured log-odds model

The basic form of the structured log-odds model introduced in section 3.1 is quite simple. The model gives a probabilistic prediction on *binary* match outcomes. The model only takes information from historical *binary* match outcomes, and the only feature it use to make prediction is the team identity. In this section, we introduce extensions of the structured log-odds model in three different aspects. These extensions can be combined together and form more sophisticated models. We will first introduce an extension that makes prediction on *ternary* match outcomes. Then, we show that the structured log-odds model can incorporate more features than just the team identity. Finally, we show how historical final scores can help the model to make more informed prediction.

3.2.1 Modeling ternary outcomes

This section addresses the issue of modeling draws raised in 2.5.3. When it is necessary to model draws, we assume that the outcome of a match is an ordinal random variable of three levels: win $\succ$ draw $\succ$ lose. The draw is treated as a middle outcome. The extension of structured log-odds model is inspired by an extension of logistic regression: the Proportional Odds model.

The Proportional Odds model is a well-known family of models for ordinal random variables (McCullagh, 1980). It extends the logistic regression to model ordinary target variables. The model parametrizes the logit transformation of the cumulative probability as a linear function of feature X_{ij}. The coefficients associated with feature variables are shared across all levels, but there is an intercept term α_k which is specific to a certain level.

$$\log(\frac{P(Y_{ij} \succ k)}{P(Y_{ij} \preceq k)}) = \alpha_k + \beta^\top X_{ij}$$

The model is called Proportional Odds model because the odds for any two different levels k_1, k_2 are proportional to each other independent of the features.

$$\left(\frac{P(Y_{ij} \succ k_1)}{P(Y_{ij} \preceq k_1)}\right) / \left(\frac{P(Y_{ij} \succ k_2)}{P(Y_{ij} \preceq k_2)}\right) = \exp(\alpha_{k_1} - \alpha_{k_2})$$

Using a similar formulation, the structured log-odds model can be extended to model draws. Let

$$\begin{aligned}
\log\left(\frac{P(Y_{ij} = \text{win})}{P(Y_{ij} = \text{draw}) + P(Y_{ij} = \text{lose})}\right) &= L_{ij} \\
\log\left(\frac{P(Y_{ij} = \text{draw}) + P(Y_{ij} = \text{win})}{P(Y_{ij} = \text{lose})}\right) &= L_{ij} + \phi
\end{aligned}$$

where L_{ij} is the entry in structured log-odds matrix and ϕ is a free parameter that affects the estimated probability of a draw. Under this formulation, the probabilities for different outcomes are given by

$$\begin{aligned}
P(Y_{ij} = \text{win}) &= \sigma(L_{ij}) \\
P(Y_{ij} = \text{lose}) &= \sigma(-L_{ij} - \phi) \\
P(Y_{ij} = \text{draw}) &= \sigma(-L_{ij}) - \sigma(-L_{ij} - \phi)
\end{aligned}$$

3.2.2 The structured log-odds model with features

Up till now, the structured log-odds matrix is solely determined by unknown factors that reflect the strength of the team. Usually, features related to the match outcomes are also available. This issue is raised in the context of the Élő model in section 2.5.3. The structured log-odds model can be extended to incorporate additional features. The extension is essentially a combination of the structured log-odds model and logistic regression.

As illustrated in equation 18, the log-odds in this extension is the sum of two components: an entry in the structured log-odds matrix and a feature-dependent component.

$$L_{ij}^{(K)} = L_{ij} + \beta^{\top} x_{ij} \tag{18}$$

In principle, one can add any additional covariates as long as the log-odds matrix is identifiable. The general form of log-odds matrix with K additional features is

$$L^{(K)} = L + \sum_{i=1}^{K} \beta_k \cdot C_k$$

where β's are coefficients and C's are matrices holding features.

A common assumption in competitive team sports is that the home team has a better chance of winning regardless of which opponent it is playing against. The extension allows us to add this piece of information into the log-odds matrix

$$L' = L + \beta \cdot H$$

β is a free parameter. The covariate matrix H specifies which team is the home team. $H_{ij} = 1$ if $i \neq j$ and $H_{ij} = 0$ if $i = j$. In this way, the (i, j) entry in the new matrix L' is the log-odds that team i wins team j on its home field. Also note that the new matrix L' is no longer antisymmetric.

3.2.3 Incorporating historical final scores

In section 2.5.3, we introduced several existing methods that use score differences to update the Élő ratings. In this section, we derive a principled way to incorporate such information into the model.

A closer look into the structured log-odds model suggests that the model has two main components: a structured matrix that contains parameters, and a distribution that links parameters to observations. The basic structured log-odds model utilizes binary match outcomes, and naturally, the linking distribution is chosen to be the Binomial distribution. Using a suitable linking distribution, the model can utilize additional information in final scores.

The Skellam's distribution models the difference between two Poisson distributions (Skellam, 1945). The support of Skellam's distribution is the set of integers. If Y_1 and Y_2 follow Poisson distribution with mean μ_1 and μ_2 respectively and their correlation is ρ, then their difference $\tilde{Y} = Y_1 - Y_2$ follows Skellam's distribution with parameters $\lambda_1 = \mu_1 - \rho\sqrt{\mu_1\mu_2}$ and $\lambda_2 = \mu_2 - \rho\sqrt{\mu_1\mu_2}$. The probability mass function takes two positive parameters λ_1 and λ_2 and is given by

$$P(X = x|\lambda_1, \lambda_2) = e^{-(\lambda_1+\lambda_2)} \left(\frac{\lambda_1}{\lambda_2}\right)^{x/2} I_{|z|}(2\sqrt{\lambda_1\lambda_2})$$

where $I_r(x)$ is the modified Bessel function of order r given by

$$I_r(x) = \left(\frac{x}{2}\right)^r \sum_{k=0}^{\infty} \frac{\left(\frac{x^2}{4}\right)^k}{k!\Gamma(r+k+1)}$$

Karlis and Ntzoufras (2009) first proposed to model the final score differences as a random variable following the Skellam's distribution. The proposal is reasonable if we assume that the final scores follow (potentially correlated) Poisson distributions. As we have already seen in section 2.6, the Poisson assumption is very common in the literature (Maher, 1982; Dixon and Coles, 1997; Crowder et al., 2002).

Now we are ready to extend the structured log-odds model to incorporate historical final scores. We will use Skellam's distribution as the linking distribution: we assume that the score difference of a match between team i and team j, $\tilde{Y}_{ij}$ follows a Skellam's distribution with parameter $\lambda_1 = \Lambda_{ij}$ and $\lambda_2 = \Lambda_{ji}$. The matrix Λ is the structured matrix containing free parameters Λ_{ij}. For example, We can assume that the matrix Λ has rank one.

$$\Lambda = uv^\top$$

where u and v are two vectors with all elements greater than zero. Note that we can still add additional features such as home advantage to the structured parameter matrix Λ using the way introduced in section 3.2.2.

The prediction for ternary match outcomes can be derived from predicted score difference $\tilde{Y}_{ij}$. In contrast to section 3.2.1, the probability of draw can now be calculated without introducing any additional parameter.

$$\begin{aligned} P(\text{win}) &= P(\tilde{Y}_{ij} > 0) \\ P(\text{draw}) &= P(\tilde{Y}_{ij} = 0) \\ P(\text{lose}) &= P(\tilde{Y}_{ij} < 0) \end{aligned}$$

3.3 Training the structured log-odds model and its extensions

In this section, we introduce three training methods for structured log-odds models. The methods are generic and they can be applied to all variants introduced so far.

A general way of estimating the parameters is the Maximum Likelihood procedure. As mentioned in section 3.1, the log-likelihood function ℓ of all structured log-odds models for binary match outcome has the same form (15). These include models with different structural assumptions and models that use additional features. As a result, the derivative of the log-likelihood also has the same form (19). In this equation, θ may refer to any parameter in the model such as an unknown factor or a coefficient associated with a feature. The equation shows that for different model variants the only difference occurs in the gradient term $\frac{\partial}{\partial \theta_i} L_{ij}$. This fact enables us to develop a unified training method for a variety of models.

$$\begin{aligned}
\frac{\partial}{\partial \theta_i}\ell &= \frac{\partial}{\partial \theta_i}\left\{\sum_{t=1}^{T}\left[Y_{ij}(t)\log p_{ij} + \big(1-Y_{ij}(t)\big)\log(1-p_{ij})\right]\right\} \\
&= \sum_{t=1}^{T}\left\{\left[Y_{ij}(t)(1-p_{ij}) - \big(1-Y_{ij}(t)\big)\,p_{ij}\right]\cdot\frac{\partial}{\partial \theta_i}\log p_{ij}\right\} \qquad (19)\\
&= \sum_{t=1}^{T}\left\{\left[Y_{ij}(t)-p_{ij}\right]\cdot\frac{\partial}{\partial \theta_i}L_{ij}\right\}
\end{aligned}$$

The extensions for ternary match outcome and score difference have a similar property but the log-likelihood function is not the same as (15). All training methods introduced in this section are still applicable to them.

3.3.1 The online training method

If we use online Gradient Ascent method with mini-batch size one and learning rate α to maximize of log-likelihood (15), the update rule at time step t will be

$$\theta_i(t+1) \leftarrow \theta_i(t) + \alpha\left[Y_{ij}(t)-p_{ij}\right]\cdot\frac{\partial}{\partial \theta_i}L_{ij} \qquad (20)$$

For the Élő model, the derivative $\frac{\partial}{\partial \theta_i}\log p_{ij} = 1$ and the update rule reduces to $\theta_i \leftarrow \theta_i + \alpha\left[Y_{ij}-p_{ij}\right]$. Comparing this update rule with equation 8, we can immediately observe that the two updates are equivalent. The hyper-parameter "K factor" in the Élő model in fact corresponds to the learning rate in the Stochastic Gradient Ascent update. It is worth emphasizing that the parameters found by this update rule may not correspond to the maximum likelihood estimates since the old samples are discarded after being used only once.

For other variants, update rule (20) is still applicable. We only need to specify the model-dependent derivative term $\frac{\partial}{\partial \theta_i} L_{ij}$ in the formula.

The algorithm is summarized in (1). $\{x_{ij}\}$ and $\{y_{ij}\}$ contains the sequence of features and match outcomes. At each time, the update method of the model takes the new feature and outcome, and updates the parameters according to equation (20).

Algorithm 1 Online training method

Require: $\{x_{ij}\}$, $\{y_{ij}\}$, model, K-factor
  for $t \leftarrow 0 : T$ **do**
    model.update($x_{ij}[t]$, $y_{ij}[t]$)
  end for

The online updating rule has two obvious advantages. Firstly, the update rule has a simple formula. The update of the parameters is therefore computationally efficient. Secondly, by fixing the learning rate as a constant, the model would emphasize on more recent matches. It is widely hypothesized that the team's performance changes gradually over time. The online update with fixed learning rate can accommodate this non-stationarity.

The downside of the online update is that it does not utilize the samples efficiently. The match outcomes are only used once during the update. Therefore, the parameters may never maximize the log-likelihood function at any given time. When more parameters are introduced into the model, the landscape of the log-likelihood function may be more complex, and the optimization can be much harder. The online update may require a large number of samples to obtain a sensible estimate of the parameters for rank-four Élő model or the model with covariates.

Another issue about the online update rule is that updating the home advantage and the coefficients of other features can be counter-intuitive because we expect these coefficients to keep constant. In the original Élő model, the home advantage is treated as a hyper-parameter. Similarly, we will treat other coefficients as hyper-parameters and tune them by a grid search using the evaluation data set. However, these parameters can also be regarded as ordinary parameters and estimated by maximizing the log-likelihood function using batch optimization. In this way, we save the computation of the grid search and obtain a more precise estimation.

3.3.2 The batch training method

It is also possible to perform batch optimization on log-likelihood function 15 directly. Essentially, this is equivalent to solving equation (21) for all unknown parameters. Since the derivative of log-likelihood has an explicit form, we can use the batch Gradient Ascent method. In this way, the parameters will correspond to the true maximum likelihood estimates under independence assumption. An additional advantage of the batch training method is that we no longer need the

hyper-parameter "K factor".

$$\sum_{t=1}^{T} \left\{ \left[Y_{ij}(t) - p_{ij} \right] \cdot \frac{\partial}{\partial \theta_i} L_{ij} \right\} = 0 \tag{21}$$

In practice, the training data accumulate through time, so we need to re-train the model periodically in order to capture new information.

The batch training algorithm is summarized in (2). Every Δt time, the model evokes the train method on all historical features and outcomes. The parameters are updated by solving (21). The model can then be used for prediction before the next re-training happens.

Algorithm 2 Batch training method

Require: $\{x_{ij}\}$, $\{y_{ij}\}$, model, Δt
  $t = 0$
  while $t < T$ **do**
    model.train$(x_{ij}[0:t],\ y_{ij}[0:t])$
    use model to predict next Δt matches
    $t \leftarrow t + \Delta t$
  end while

3.3.3 The two-stage training method

The batch method poses a significant computational overhead in the retraining step while the online update does not utilize data efficiently. Judging from these two factors, we propose a two-stage method that combines batch training and online updates for the structured log-odds model.

In the first stage, the estimate of all parameters including home advantage and coefficients for other covariates are obtained by maximizing the log-likelihood function ℓ **??** on a fixed training data set using batch optimization. This gives us a good initial value for all parameters.

In the second stage (typically in the testing phase), the team-specific rating factors are updated using the online approach while home advantage and other coefficients are fixed. This online training updates the model with the latest match outcomes.

We also need to specify the hyper-parameter "K factor" when using the two-stage method.

The algorithm is summarized in 3. The model first evokes batch training for match outcomes and features up to t_0. After that, the model is switched to online training mode.

3.4 Regularized log-odds matrix estimation

All the structured log-odds models we discussed so far made explicit assumption about the structure of the log-odds matrix. An alternative way is to encourage the log-odds matrix to be more

Algorithm 3 Two-stage training method

Require: $\{x_{ij}\}$, $\{y_{ij}\}$, model, t_0, K-factor
  model.train($x_{ij}[0:t_0]$, $y_{ij}[0:t_0]$)
  for $t \leftarrow t_0 : T$ **do**
    model.update($x_{ij}[t]$, $y_{ij}[t]$)
  end for

structured by imposing an penalty on its complexity. In this way, there is no need to specify the structure explicitly. The trade-off between the log-odds matrix's complexity and its ability to explain observed data is tuned by validation on evaluation data set.

Let us recall the independent Bernoulli assumption made at the beginning of section 3.1. If we do not make any assumption about the structure of winning probabilities, the maximum likelihood estimate for each p_{ij} is given by

$$\hat{p}_{ij} = \frac{w_{ij}}{N_{ij}}$$

where w_{ij} is the number of matches in which team i beats team j, and N_{ij} is the total number of matches between team i and team j. If we parametrize the distribution with log-odds L_{ij}, we obtain its maximum likelihood estimate as follows:

$$\hat{L}_{ij} = \log\left(\frac{\hat{p}_{ij}}{1-\hat{p}_{ij}}\right)$$

This follows from the fact that the maximum likelihood estimates are invariant under re-parametrization. We will call the matrix $\hat{L}$ the empirical log-odds matrix. It is worth noticing that the empirical log-odds matrix gives the best explanation for observed data because it maximizes the likelihood function. Any structured log-odds matrix will achieve a lower likelihood on observed data. However, in practice the empirical log-odds matrix often has very poor predictive performance because the estimate tends to have very large variance.

By imposing a complexity penalty, the new model reduces the variance of the estimated log-odds matrix. Common complexity measures of a matrix are its matrix norms Srebro and Shraibman (2005). In recent years, the trace norm of a matrix has attracted much attention. It has find a wide range of machine-learning applications including matrix completion (Candès and Recht, 2009), matrix factorization (Srebro et al., 2004), and multi-task learning (Pong et al., 2010). In particular, the trace norm of a matrix L is defined as

$$||L||_* = \sum_{k=1}^{N} \sigma_k(L)$$

where $\sigma_k(L)$ is the k^{th} singular value of matrix L. The trace norm of a matrix is closely related to

the matrix's rank since the rank is the number of non-zero singular values. In practice, the trace norm is often used as a surrogate for the rank in order to make the optimization easier.

The optimization problem for regularized log-odds matrix estimation can be formulated as

$$L^* = \arg\min \left\{ ||\hat{L} - L||_F^2 + \lambda ||L||_* \right\}$$

subject to

$$L + L^\top = 0_{n \times n}$$

The "error term" is chosen to be the squared loss: $\sum_i \sum_j (L_{ij} - \hat{L}_{ij})^2$ instead of the log-likelihood function in order to make the objective function a quadratic function of parameters. The quadratic function is usually easier to optimize.

There is a well-known bound on the trace of a matrix (Srebro, 2004): For any $X \in \mathbb{R}^{n \times m}$, and $t \in \mathbb{R}$, $||X||_* \leq t$ if and only if there exists $A \in \mathbb{S}^n$ and $B \in \mathbb{S}^m$ such that $\begin{bmatrix} A & X \\ X^\top & B \end{bmatrix} \succeq 0$ and $\frac{1}{2}(tr(A) + tr(B)) < t$. Utilizing this bound, we can introduce two auxiliary matrices A and B and solve an equivalent problem:

$$\min_{A,B,L} \left\{ ||\hat{L} - L||_F^2 + \frac{\lambda}{2}(tr(A) + tr(B)) \right\}$$

subject to

$$\begin{bmatrix} A & L \\ L^\top & B \end{bmatrix} \succeq 0$$

and

$$L + L^\top = 0_{n \times n}$$

This is a Quadratic Program with a positive semidefinite constraint and a linear equality constraint. It can be efficiently solved by the interior point method Vandenberghe and Boyd (1996), and alternative algorithms for large scale problem also exist (Mishra et al., 2013).

The regularized log-odds matrix estimation method is quite experimental as we have not established a mathematical proof for the error bound. It is also quite restrictive as it cannot model the possibility of draws or use additional features. Further research is also needed to find an online update formula for this method.

4 Experiments

4.1 Synthetic data

In this section, we present the experiment results over synthetic data sets. The goal of these experiments is to show that the newly proposed structured log-odds models perform better than the original Élő model when the data were generated following the new models' assumptions. The experiments also show the validity of the parameter estimation procedure.

The synthetic data are generated according to the assumptions of the structured log-odds models (13). To recap, the data generation procedure is the following.

1. The binary match outcome y_{ij} is sampled from a Bernoulli distribution with success probability p_{ij},

2. The corresponding log-odds matrix L has a certain structure,

3. The match outcomes are sampled independently (there is no temporal effect)

As the first step in the procedure, we randomly generate a ground truth log-odds matrix with a certain structure. The structure depends on the model in question and the matrix generation procedure is different for different experiments. The match outcomes y_{ij}'s are sampled independently from the corresponding Bernoulli random variables with success probabilities p_{ij} derived from the true log-odds matrix.

For a given ground truth matrix, we generate a validation set and an independent test set in order to tune the hyper-parameter. The hyper-parameters are the *K factor* for the structured log-odds models, and the *regularizing strength* λ for regularized log-odds matrix estimation. We perform a grid search to tune the hyper-parameter. We choose the hyper-parameter to be the one that achieves the best log-likelihood on the validation set. The model with the selected hyper-parameter is then evaluated on the test set. This validation setting is sound because of the independence assumption (3).

The tuned model gives a probabilistic prediction for each match in the test set. Based on these predictions, we can calculate the mean log-likelihood or the mean accuracy on the test set. If two models are evaluated on the same test set, the evaluation metrics for the two models form a paired sample. This is because the metrics depend on the specific test set.

In each experiment, we replicate the above procedure for many times. In each replication, a new ground truth log-odds matrix is generated, and the models are tuned and evaluated. Each replication hence produces a paired sample of evaluation metrics because the metrics for different models are conditional independent in the same replication.

We would like to know which model performs better given the data generation procedure. This question can be answered by performing hypothesis testing on paired evaluation metrics produced by the replications. We will use the paired Wilcoxon test because of the violation of normality assumption.

The experiments do not aim at comparing different training methods (3.3). Hence, all models in an experiment are trained using the same method to enable an apple-to-apple comparison. In experiments 4.1.1 and 4.1.2, the structured log-odds models and the Élő model are trained by the online update algorithm. Experiment (4.1.3) concerns about the regularized log-odds matrix estimation, whose online update algorithm is yet to be derived. Therefore, all models in section 4.1.3 are trained using batch training method.

The experiments all involve 47 teams [3]. Both validation and test set include four matches between each pair of teams.

4.1.1 Two-factor Élő model

This experiment is designed to show that the two-factor Élő model is superior to the Élő model if the true log-odds matrix is a general rank-two matrix.

Components in the two factors u and v are independently generated from a Gaussian distribution with $\mu = 1$ and $\sigma = 0.7$. The true log-odds matrix is calculated as in equation 16 using the generated factors. The rest of the procedure is carried out as described in section 4.1. This procedure is repeated for two hundred times.

The two hundred samples of paired mean accuracy and paired mean log-likelihood are visualized in figure 1 and 2. Each point represents an independent paired sample.

Our hypothesis is that if the true log-odds matrix is a general rank-two matrix, the two-factor Élő model is likely to perform better than the original Élő model. We perform Wilcoxon test on the paired samples obtained in the experiments. The two-factor Élő model produces significantly better results in both metrics (one-sided p-value is 0.046 for accuracy and less than 2^{-16} for mean log-likelihood).

[3] Forty-seven teams played in the English Premier league between 1993 and 2015

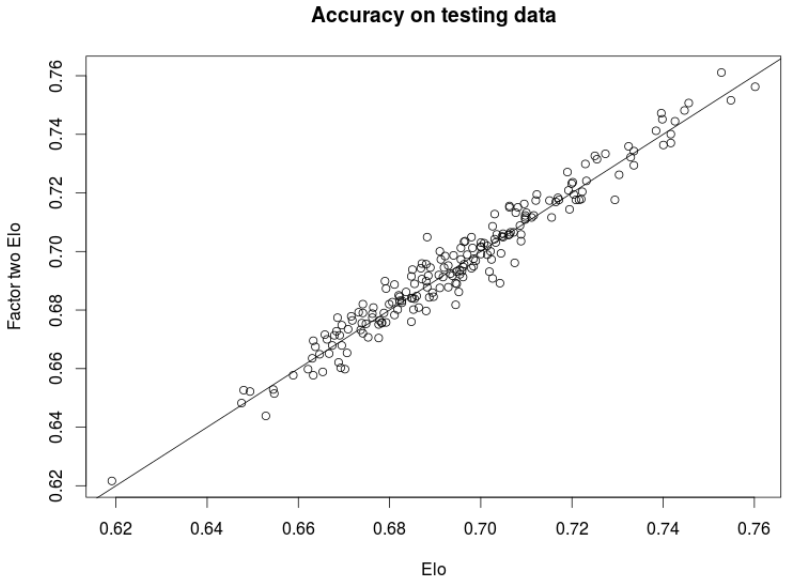

Figure 1: Each dot represents the testing accuracy in an experiment. The X-asis shows the accuracy achieved by the Élő model while the Y-axis shows the accuracy achieved by the two-factor Élő.

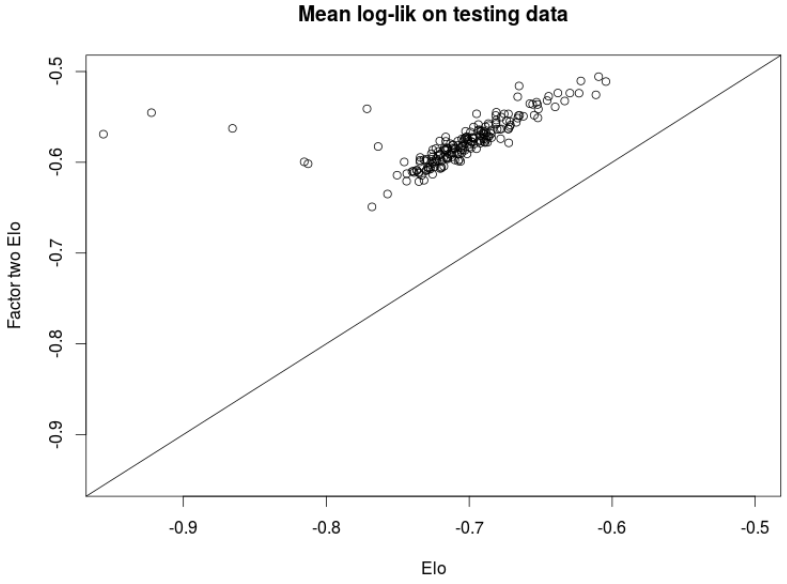

Figure 2: Each dot represents the mean log-likelihood on testing data in an experiment. The X-asis shows the mean log-likelihood achieved by the Élő model while the Y-axis shows the mean log-likelihood achieved by the two-factor Élő.

4.1.2 Rank-four Élő model

These two experiments are designed to compare the rank-four Élő model to the two-factor Élő model when the true log-odds matrix is a rank-four matrix.

The first experiment considers the scenario when all singular values of the true log-odds matrix are big. In this case, the best rank-two approximation to the true log-odds matrix will give a relatively large error because the third and fourth singular components cannot be recovered. The log-odds matrices considered in this experiment takes the following form

$$L = s_1 \cdot u \cdot v^\top + s_2 \cdot \theta \cdot \underline{1}^\top - s_1 \cdot v \cdot u^\top - s_2 \cdot \underline{1} \cdot \theta^\top \tag{22}$$

, where s_1 and s_2 are the two distinct singular values and $\underline{1}$ is parallel to the vector of ones, and vector $\underline{1}$, u, v and θ are orthonormal. This formulation is based on the decomposition of a real antisymmetric matrix stated in section 3.1.1. The true log-odds matrix L has four non-zero singular values s_1, $-s_1$, s_2 and $-s_2$. In the experiment, $s_1 = 25$ and $s_2 = 24$.

The rest of the data generation and validation setting is the same as the experiments in section 2. The procedure is repeated for 100 times. We applied the paired Wilcoxon test to the 100 paired evaluation results. The test results support the hypothesis that the rank-four Élő model performs significantly better in both metrics (one-sided p-value is less than 2^{-16} for both accuracy and mean log-likelihood).

In the second experiment, the components in factors u, v and θ are independently generated from a Gaussian distribution with $\mu = 1$ and $\sigma = 0.7$. The log-odds matrix is then calculated using equation 17 directly. The factors are no longer orthogonal and the second pair of singular values are often much smaller than the first pair. In this case, the best rank-two approximation will be close to the true log-odds matrix.

The procedure is repeated for 100 times again using the same data generation and validation setting. Paired Wilcoxon test shows rank-four Élő model achieves significantly higher accuracy on the test data (one-sided p-value is 0.015), but the mean log-likelihood is not significantly different (p-value is 0.81).

The results of the above two experiments suggest that the rank-four Élő model will have significantly better performance when the true log-odds matrix has rank four and it cannot be approximated well by a rank-two matrix.

4.1.3 Regularized log-odds matrix estimation

In the following two experiments, we want to compare the regularized log-odds matrix estimation method with various structured log-odds models.

To carry out regularized log-odds matrix estimation, we need to first get an empirical estimate of log-odds on the training set. Since there are only four matches between any pair of teams in the training data, the estimate of log-odds often turn out to be infinity due to division by zero. Therefore, I introduced a small regularization term in the estimation of empirical winning probability $\hat{p} = \frac{n_{win}+\varepsilon}{n_{total}+2\varepsilon}$, where ε is set to be 0.01. Then, we obtain the smoothed log-odds matrix by solving the optimization problem described in section 3.4. A sequence of λ's are fitted, and the best one is chosen according to the log-likelihood on the evaluation set. The selected model is then evaluated on the testing data set.

Structured log-odds models with different structural assumptions are used for comparison. We consider the Élő model, two-factor Élő model, and rank-four Élő model. For each of the three models, we first tune the hyper-parameter on a further split of training data. Then, we evaluate the models with the best hyper-parameter on the evaluation set and select the best model. Finally, we test the selected model on the test set to produce evaluation metrics. This experiment setting imitates the real application where we need to select the model with best structural assumption.

In order to compare fairly with the trace norm regularization method (which is currently a batch method), the structured log-odds models are trained with batch method and the selected model is not updated during testing.

In the first experiment, it is assumed that the structure of log-odds matrix follows the assumption of the rank-four Élő model. The log-odds matrix is generated using equation (22) with $s_1 = 25$ and $s_2 = 2.5$. The data generation and hypothesis testing procedure remains the same as previous experiments. Paired Wilcoxon test is performed to examine the hypothesis that regularized log-odds model produces higher out-of-sample log-likelihood. The testing result is in favour of this hypothesis (p-value is less than 10^{-10}).

In the second experiment, it is assumed that the structure of log-odds matrix follows the assumption of the Élő model (section 2.5). The true Élő ratings are generated using a normal distribution with mean 0 and standard deviation 0.8. Paired Wilcoxon test shows that the out-of-sample likelihood is somewhat different between the tuned regularized log-odds model and trace norm regularization (two sided p-value = 0.09).

The experiments show that regularized log-odds estimation can adapt to different structures of the log-odds matrix by varying the regularization parameter. The performance on simulated data set is not worse than the tuned regularized log-odds model.

4.2 Real data set

4.2.1 Description of the data set

The whole data set under investigation consists of English Premier League football matches from 1993-94 to 2014-15 season. There are 8524 matches in total. The English Premier League is chosen as a representative as competitive team sports because of its high popularity. In each season, twenty teams will compete against each other using the double round-robin system: each team plays the others twice, once at the home field and once as guest team. The winner of each match scores three championship points. If the match draws, both teams score one point. The final ranking of the teams are determined by the championship points scored in the season. The team with the highest rank will be the champion and the three teams with the lowest rank will move to Division One (a lower-division football league) next season. Similarly, three best performing teams will be promoted from Division One into the Premier League each year. In the data set, 47 teams has played in the Premier League.

The data set contains the date of the match, the home team, the away team, and the final scores for both teams. The testing data consists of matches from 2010 to 2015 (2084 matches in total). The data set for hyper-parameter tunning consists of matches from 2006 to 2010. All the remaining data form the training data. The data set is retrieved from http://www.football-data.co.uk/.

The algorithms are allowed to use all available information prior to the match to predict the outcome of the match (win, lose, draw).

4.2.2 Validation setting

In the study of the real data set, we need a proper way to quantify the predictive performance of a model. This is important for two reasons. Firstly, we need to tune the hyper-parameters in the model by performing model validation. The hyper-parameters that bring best performance will be chosen. More importantly, we wish to compare the performance of different types of models scientifically. Such comparison is impossible without a quantitative measure on model performance.

It is a well-known fact that the errors made on the training data will underestimate the model's true generalization error. The common approaches to assess the goodness of a model include cross validation and bootstrapping (Stone, 1974; Efron and Tibshirani, 1997). However, both methods assume that the data records are statistically independent. In particular, the records should not contain temporal structure. In the literature, the validation for data with temporal structure is largely an unexplored area. However, the independence assumption is plausibly violated in this study and it is highly likely to affect the result. Hence, we designed an set of ad-hoc validation methods tailored for the current application.

The validation algorithm for *batch* training method Recall from section 2.1 that we need to re-train the model periodically if we adopt the batch training method. The validation algorithm for *batch* learning methods is describe in display 4. The validation algorithm takes three arguments: a training data set, a testing data set and a model object. The training data set is always available to the model. The testing data set is indexed by time period and it is sequentially made available to the model. The model object has a batch training method model.train() and a predict method model.predict(). It is worth noticing that the training method will completely update all the parameters in the model. At the start of each time period, the algorithm would train the model using all training data as well as the testing data which the model has already made predictions on. Then the model will predict the match outcomes within the time period. The predictions for all time periods are collected and finally the evaluation metrics are calculated from these predictions.

Algorithm 4 Validation setting for batch learning methods

Require: $data\_train$, $data\_test$, model
  for $t \leftarrow 0, T$ **do**
    $model.train(data\_train + data\_test[0:t-1])$
    $prediction[t] = model.predict(data\_test[t])$
  end for
  return $evaluate(data\_test, prediction)$

In general, a good validation setting should resemble the usage of the model in practice. Validation setting 4 guarantees that no future information will be used in making current predictions. It also captures the fact that the data are aggregated through time, i.e. more data become available for later years, and that the model needs to be re-trained with all previous match data after being used for some time.

The validation setting 4 has a minor issue. The matches within one time period may still be dependent with each other. Using a shorter time period length reduces such dependence, but it also requires re-training the model more often. Balancing the two factors, we choose the time period to be *three months* in the experiments.

The validation algorithm for *online* training method The validation algorithm for *online* learning methods is slightly simpler because the model does not need to be retrained periodically. As illustrated in algorithm 5, the model first perform online learning on the training data; it then perform the same procedure on the testing data. The "predict_update" method of the model contains an implicit loop over the data set. It sequentially makes predictions on data records and update the model parameters after each prediction. The predictions on the testing data are used to calculate the evaluation metrics.

Algorithm 5 Validation setting for online learning methods

Require: $data\_train$, $data\_test$, model
  $prediction\_train = model.predict\_update(data\_train)$
  $prediction\_test = model.predict\_update(data\_test)$
  return $evaluate(data\_test, prediction\_test)$

The validation algorithm for the *two-stage* training method In section 3.3.3, we compared the properties of the online training method and the batch training method; we then proposed the new two-stage training method. The validation setting for two stage training method is similar to the one for online method. The only difference is that the model is now trained using batch method on the training data.

Algorithm 6 Validation setting for two-stage learning methods

Require: $data\_train$, $data\_test$, model
  $model.train(data\_train)$
  $prediction\_test = model.predict\_update(data\_test)$
  return $evaluate(data\_test, prediction\_test)$

Hyper-parameter selection Most models in this comparative study have tunable hyper-parameters. Those hyper-parameters are tuned using the above validation settings. The data before 2006 are used as training data ("data_train") and the data between 2006 and 2010 are used as the testing data ("data_test"). Typically, a gird search is performed and the hyper-parameter which achieves highest likelihood is chosen.

Finally, the model with the selected hyper-parameters is tested using the same validation settings. The training data now consist of all data before 2010 and the testing data consist of matches after 2010.

Comparing different training methods Later in the experiment, we will compare different training methods. To ensure a fair comparison, we have made sure that the training-evaluating-testing split is the same for all training methods. This means that the model will be accessible to the same data set regardless of what training method is being used.

A complication arises in the comparison between batch training method and the other two methods. Both online and two-stage method perform immediate update during testing, but the batch method only re-trains the model periodically. This means that the batch method has a slight disadvantage because the most recent match outcomes in the test set are not reflected in the model. One can eliminate such disadvantage by performing batch re-training after each new match outcome is observed. However, we find this approach very computationally intensive and unpractical.

Hence, in this study we only compare the online training method and the two-stage training method.

4.2.3 Quantitative comparison for the evaluation metrics

We use log-likelihood and accuracy on the testing data set as evaluation metrics. We apply statistical hypothesis testing on the validation results to compare the models quantitatively.

We calculate the log-likelihood on each test case for each model. If we are comparing two models, the evaluation metrics for each test case will form a paired sample. This is because test cases might be correlated with each other and model's performance is independent given the test case. The paired t-test is used to test whether there is a significant difference in the mean of log-likelihood. We draw independent bootstrap samples with replacement from the log-likelihood values on test cases, and calculate the mean for each sample. We then calculate the 95% confidence interval for the mean log-likelihood based on the empirical quantiles of bootstrapped means (Davison and Hinkley, 1997). Five thousand bootstrap samples are used to calculate these intervals.

The confidence interval for accuracy is constructed assuming the model's prediction for each test case, independently, has a probability p to be correct. The reported 95% confidence interval for Binomial random variable is calculated from a procedure first given in Clopper and Pearson (1934). The procedure guarantees that the confidence level is at least 95%, but it may not produce the shortest-length interval.

The models are also compared with a set of benchmarks. The first benchmark always predicts home team to win the match. The second benchmark is constructed from the betting odds given by different bookmakers. The probability implied by betting odds is used as prediction. Typically, the odds will include a vig so the implied "probability" does not sum to one. They are normalized to give the valid probability. The historical odds are also obtained from http://www.football-data.co.uk/.

4.2.4 Performance of the structured log-odds model

We performed the tunning and validation of the structured log-odds models using the method described in section 4.2.2. The following list shows all models examined by this experiment:

1. The Élő model (section 2.5)
2. Two-factor Élő model (section 3.1)
3. Rank-four Élő model (section 3.1)
4. The Élő model with score difference (section 3.2.3)

5. The Élő model with two additional features (section 3.2.2)

All models include a free parameter for home advantage (see section 3.2.2), and they are also able to capture the probability of a draw (section 3.1). We have introduced two covariates in the fifth model. These two covariates indicate whether the home team or away team is just promoted from Division One this season. We have also tested the trace norm regularized log-odds model, but as indicated in section 3.4 the model still has many limitations for the application to the real data. The validation results are summarized in table 3 and table 4. We do not report the log-likelihood for trace norm regularized log-odds model because it cannot model draws, and has negative infinity log-likelihood.

The testing results help us understand the following two scientific questions:

1. Which training method brings the best performance to structured log-odds models?
2. Which type of structured log-odds model achieves best performance on the data set?

In order to answer the first question, we test the following hypothesis:

(H1): Null hypothesis: for a certain model, two-stage training method and online training method produce the same mean out-of-sample log-likelihood. Alternative hypothesis: for a certain model two-stage training method produces a higher mean out-of-sample log-likelihood than online training method.

For the reasons discussed in section 4.2.2, we only compare the online and two-stage training method in this experiment. The paired t-test is used to assess the above hypotheses. The p-values are shown in table 2. The cell associated with the Élő model with covariates are empty because the online training method does not update the coefficients for features. The first columns of the table gives strong evidence that the two-stage training method should be preferred over online training. All tests are highly significant even if we take into account the issue of multiple testing.

In order to answer the second question, we compare the four new models with the Élő model. The hypothesis is formulated as

(H2): Null hypothesis: using the best training method, the new model and the Élő model produce the same mean out-of-sample log-likelihood. Alternative hypothesis: using the best training method, the new model produces a higher mean out-of-sample log-likelihood than the Élő model.

The p-values are listed in the last column of table 2. The result also shows that adding more factors in the model does not significantly improve the performance. Neither two-factor model nor rank-four model outperforms the original Élő model on the testing data set. This might provide evidence

and justification of using the Élő model on real data set. The model that uses the score difference performs slightly better than the original Élő model. However, the difference in out-of-sample log-likelihood is not statistically significant (the p-value for one-sided test is 0.24 for likelihood). Adding additional covariates about team promotion significantly improves the Élő model.

Type	H1	H2
Élő model	7.8×10^{-5}	-
Two-factor model	4.4×10^{-14}	~1
Rank-four model	9.8×10^{-9}	~1
Score difference	2.2×10^{-16}	0.235
Élő model with covariates	-	0.002

Table 2: Hypothesis testing on the structured log-odds model. The column "Type" specifies the type of the model; the remaining two columns shows the one-sided p-values for the associated hypothesis

Type	Method	Acc	2.5%	97.5%
Benchmark	Home team win	46.07%	43.93%	48.21%
	Bet365 odds	**54.13%**	51.96%	56.28%
Élő model	Two-stage	52.40%	50.23%	54.56%
	Online	52.16%	50.00%	54.32%
	Batch	50.58%	48.41%	52.74%
Two-factor model	Two-stage	51.30%	49.13%	53.46%
	Online	50.34%	48.17%	52.50%
	Batch	50.86%	48.69%	53.03%
Rank-four model	Two-stage	51.34%	49.17%	53.51%
	Online	50.34%	48.17%	52.50%
	Batch	50.58%	48.41%	52.74%
Score difference	Two-stage	52.59%	50.42%	54.75%
	Online	47.17%	45.01%	49.34%
	Batch	51.10%	48.93%	53.27%
Élő model with covariates	Two-stage	**52.78%**	50.61%	54.95%
	Batch	50.86%	48.69%	53.03%
Trace norm regularized model	Batch	45.89%	43.54%	48.21%

Table 3: Structured log-odds model's accuracy on testing data. The column "Type" specifies the type of the model; the column "Method" specifies the training method. Testing accuracy is given in the column "Acc". The last two columns gives the 95% confidence interval for testing accuracy

Type	Method	Mean log-lik	2.5%	97.5%
Benchmark	Bet365 odds	**-0.9669**	-0.9877	-0.9460
Élő model	Two-stage	-0.9854	-1.0074	-0.9625
	Online	-1.0003	-1.0254	-0.9754
	Batch	-1.0079	-1.0314	-0.9848
Two-factor model	Two-stage	-1.0058	-1.0286	-0.9816
	Online	-1.0870	-1.1241	-1.0504
	Batch	-1.0158	-1.0379	-0.9919
Rank-four model	Two-stage	-1.0295	-1.0574	-1.0016
	Online	-1.1024	-1.0638	-1.1421
	Batch	-1.0078	-1.0291	-0.9860
Score difference	Two-stage	-0.9828	-1.0034	-0.9623
	Online	-1.1217	-1.1593	-1.0833
	Batch	-1.0009	-1.0206	-0.9802
Élő model with covariates	Two-stage	**-0.9807**	-1.0016	-0.9599
	Batch	-1.0002	-1.0204	-0.9798

Table 4: Structured log-odds model's mean log-likelihood on testing data. The column "Type" specifies the type of the model; the column "Method" specifies the training method. Mean out-of-sample log-likelihood is given in the column "Mean log-lik". The last two columns gives the 95% confidence interval for mean out-of-sample log-likelihood

4.2.5 Performance of the batch learning models

This experiment compare the performance of batch learning models. The following list shows all models examined by this experiment:

1. GLM with elastic net penalty using multinomial link function
2. GLM with elastic net penalty using ordinal link function
3. Random forest
4. Dixcon-Coles model

The first three models are machine learning models that can be trained on different features. The following features are considered in this experiment:

1. Team id: the identity of home team and away team
2. Ranking: the team's current ranking in Championship points and goals
3. VS: the percentage of time that home team beats away team in last 3, 6, and 9 matches between them

4. Moving average: the moving average of the following monthly features using lag 3, 6, 12, and 24

   (a) percentage of winning at home
   (b) percentage of winning away
   (c) number of matches at home
   (d) number of matches away
   (e) championship points earned
   (f) number of goals won at home
   (g) number of goals won away
   (h) number of goals conceded at home
   (i) number of goals conceded away

The testing accuracy and out-of-sample log-likelihood are summarized in table 8 and table 9. All models perform better than the baseline benchmark, but no model seems to outperform the state-of-the-art benchmark (betting odds).

We applied statistical testing to understand the following questions

1. Does the GLM with ordinal link function perform better than the GLM with multinomial link function?
2. Which set of features are most useful to make prediction?
3. Which model performs best among GLM, Random forest, and Dixcon-Coles model?

For question one, we formulate the hypothesis as:

(H3): Null hypothesis: for a given set of feature, the GLM with ordinal link function and the GLM with multinomial link function produce the same mean out-of-sample log-likelihood. Alternative hypothesis: for a given set of feature, the mean out-of-sample log-likelihood is different for the two models.

The p-values for these tests are summarized in table 5. In three out of four scenarios, the test is not significant. There does not seem to be enough evidence against the null hypothesis. Hence, we retain our believe that the GLM with different link functions have the same performance in terms of mean out-of-sample log-likelihood.

For question two, we observe that models with the moving average feature have achieved better performance than the same model trained with other features. We formulate the hypothesis as:

Features	p-value
Team_id only	0.148
Team_id and ranking	0.035
Team_id and VS	0.118
Team_id and MA	0.121

Table 5: p-values for H3

Features	GLM1	GLM2
Team_id only	2.7×10^{-12}	5.3×10^{-8}
Team_id and ranking	1.2×10^{-9}	3.7×10^{-6}
Team_id and VS	0.044	0.004

Table 6: p-values for H4: the column "Features" are the alternative features compared with the moving average features. The next two columns contain the p-values for the GLM with multinomial link function (GLM1) and the GLM with ordinal link function (GLM2)

(H4): Null hypothesis: for a given model, the moving average feature and an alternative feature set produce the same mean out-of-sample log-likelihood. Alternative hypothesis: for a given model, the mean out-of-sample log-likelihood is higher for the moving average feature.

The p-values are summarized in table 6. The tests support our believe that the moving average feature set is the most useful one among those examined in this experiment.

Finally, we perform comparison among different models. The comparisons are made between the GLM with multinomial link function, Random forest, and Dixcon-Coles model. The features used are the moving average feature set. The p-values are summarized in table 7. The tests detect a significant difference between GLM and Random forest, but the other two pairs are not significantly different. We apply the p-value adjustment using Holm's method in order to control family-wise type-one error (Sinclair et al., 2013). The adjusted p-values are not significant. Hence, we retain our belief that the three models have the same predictive performance in terms of mean out-of-sample log-likelihood.

Comparison	p-value	adjusted
GLM and RF	0.03	0.08
GLM and DC	0.48	0.96
DC and RF	0.54	0.96

Table 7: p-values for model comparison: the column "Comparison" specifies which two models are being compared. "RF" stands for Random forest; "DC" stands for the Dixon-Cole model. The column "p-value" contains the two-sided p-value of the corresponding paired t-test. The column "adjusted" shows the adjusted p-values for multiple testing

Models	Features	Acc	2.5%	97.5%
Benchmark	Home team win	46.07%	43.93%	48.21%
	Bet365 odds	54.13%	51.96%	56.28%
GLM1	Team_id only	50.05%	47.88%	52.22%
	Team_id and ranking	50.62%	48.45%	52.79%
	Team_id and VS	51.25%	49.08%	53.41%
	Team_id and MA	**52.69%**	50.52%	54.85%
GLM2	Team_id only	50.67%	48.52%	52.82%
	Team_id and ranking	50.24%	48.09%	52.38%
	Team_id and VS	51.92%	49.75%	54.08%
	Team_id and MA	52.93%	50.76%	55.09%
RF	Team_id and MA	52.06%	49.89%	54.23%
Dixcon-Coles	-	52.54%	50.40%	54.68%

Table 8: Testing accuracy for batch learning models: The column "Type" specifies the type of the model; "GLM1" refers to the GLM with multinomial link function, and "GLM2" refers to the GLM with ordinal link function. column "Models" specifies the model, and the column "Features" specifies the features used to train the model. Testing accuracy is given in the column "Acc". The last two columns gives the 95% confidence interval for testing accuracy.

Models	Features	Mean log-lik	2.5%	97.5%
Benchmark	Bet365 odds	-0.9669	-0.9877	-0.9460
GLM1	Team_id only	-1.0123	-1.0296	-0.9952
	Team_id and ranking	-1.0006	-1.0175	-0.9829
	Team_id and VS	-0.9969	-1.0225	-0.9721
	Team_id and MA	**-0.9797**	-0.9993	-0.9609
GLM2	Team_id only	-1.0184	-1.0399	-0.9964
	Team_id and ranking	-1.0097	-1.0317	-0.9874
	Team_id and VS	-1.0077	-1.0338	-0.9813
	Team_id and MA	-0.9838	-1.0028	-0.9656
RF	Team_id and MA	-0.9885	-1.0090	-0.9683
Dixcon-Coles	-	-0.9842	-1.0076	-0.9610

Table 9: out-of-sample log-likelihood for batch learning models: The column "Type" specifies the type of the model; "GLM1" refers to the GLM with multinomial link function, and "GLM2" refers to the GLM with ordinal link function. the column "Models" specifies the model, and the column "Features" specifies the features used to train the model. Mean out-of-sample log-likelihood is given in the column "Mean log-lik ". The last two columns gives the 95% confidence interval for mean out-of-sample log-likelihood.

5 Summary and Conclusion

In this study, we focus on the scientific question of predicting the outcome of competitive team sports. We present a comprehensive literature review and articulate the problem mathematically. We address the issues arise in prediction and validation, and propose methods to resolve them.

The main contribution of this study is the proposal of the structured log-odds model, which provides a unified framework for several well-known models such as the Élő model and logistic regression. The unification and abstraction enable us to design extensions for the existing models. We also propose different training methods.

Another contribution is the application to the English premier league data. In particular, we systematically evaluate several existing models as well as the newly proposed model with the English premier league data. Our finding suggests that both domain-specific parametric models and feature-based machine learning predictors achieve similar level of predictive performance when the available information only includes previous match outcomes. We also find that the Élő model has better predictive power than higher-rank models in real data. Furthermore, the two-stage training method often increase the performance of the model compared to the online method. Last but equally importantly, we find that features about league promotion significantly increases the predictive performance.

Further research can study the method of regularized log-odds matrix estimation especially the way of performing online update and modeling draws. Another area of further research is the Bayesian approach to the structured log-odds model. Finally, we hope that as more features about competitive team sports become publicly available researchers can propose and validate more complex models.

References

Baio, G. and Blangiardo, M. (2010). Bayesian hierarchical model for the prediction of football results. *Journal of Applied Statistics*, 37(2):253–264.

Blythe, D. A. and Király, F. J. (2015). Prediction and quantification of individual athletic performance. *arXiv preprint arXiv:1505.01147*.

Candès, E. J. and Recht, B. (2009). Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772.

Clopper, C. J. and Pearson, E. S. (1934). The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26(4):404–413.

Constantinou, A. C., Fenton, N. E., and Neil, M. (2012). pi-football: A bayesian network model for forecasting association football match outcomes. *Knowledge-Based Systems*, 36:322–339.

Crowder, M., Dixon, M., Ledford, A., and Robinson, M. (2002). Dynamic modelling and prediction of english football league matches for betting. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 51(2):157–168.

Davison, A. C. and Hinkley, D. V. (1997). *Bootstrap methods and their application*, volume 1. Cambridge university press.

Dixon, M. J. and Coles, S. G. (1997). Modelling association football scores and inefficiencies in the football betting market. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 46(2):265–280.

Efron, B. and Tibshirani, R. (1997). Improvements on cross-validation: the 632+ bootstrap method. *Journal of the American Statistical Association*, 92(438):548–560.

Élő, A. E. (1978). *The rating of chessplayers, past and present*. Arco Pub.

Eves, H. W. (1980). *Elementary matrix theory*. Courier Corporation.

FIFA (2016). Fifa/coca-cola world ranking: Women's ranking procedure. `http://www.fifa.com/fifa-world-ranking/procedure/women.html`. Accessed: 2016-05-30.

Fleig, G. (2012). Manchester city analytics data release. `http://www.mcfc.co.uk/mcfcanalytics`. Accessed: 2016-04-06.

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

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

Goddard, J. (2005). Regression models for forecasting goals and match results in association football. *International Journal of forecasting*, 21(2):331–340.

Griffith, R. M. (1949). Odds adjustments by american horse-race bettors. *The American Journal of Psychology*, 62(2):290–294.

Hucaljuk, J. and Rakipović, A. (2011). Predicting football scores using machine learning techniques. In *MIPRO, 2011 Proceedings of the 34th International Convention*, pages 1623–1627. IEEE.

Isaacs, R. (1953). Optimal horse race bets. *The American Mathematical Monthly*, 60(5):310–315.

Jewson, S. (2004). The problem with the brier score. *arXiv preprint physics/0401046*.

Johnson, N. L., Kotz, S., and Balakrishnan, N. (1997). *Discrete multivariate distributions*, volume 165. Wiley New York.

Kampakis, S. and Adamides, A. (2014). Using twitter to predict football outcomes. *arXiv preprint arXiv:1411.1243*.

Karlis, D. and Ntzoufras, I. (2003). Analysis of sports data by using bivariate poisson models. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 52(3):381–393.

Karlis, D. and Ntzoufras, I. (2009). Bayesian modelling of football outcomes: using the skellam's distribution for the goal difference. *IMA Journal of Management Mathematics*, 20(2):133–145.

Kocherlakota, S. and Kocherlakota, K. (1992). *Bivariate discrete distributions*. Wiley Online Library.

Lasek, J., Szlávik, Z., and Bhulai, S. (2013). The predictive power of ranking systems in association football. *International Journal of Applied Pattern Recognition*, 1(1):27–46.

League, T. P. (2016). About the premier league. `http://www.premierleague.com/en-gb/about/the-worlds-most-watched-league.html`. Accessed: 2016-06-26.

Liu, B. and Lai, P. (2010). Beating the ncaa football point spread.

Maher, M. J. (1982). Modelling association football scores. *Statistica Neerlandica*, 36(3):109–118.

McCullagh, P. (1980). Regression models for ordinal data. *Journal of the royal statistical society. Series B (Methodological)*, pages 109–142.

Mishra, B., Meyer, G., Bach, F., and Sepulchre, R. (2013). Low-rank optimization with trace norm penalty. *SIAM Journal on Optimization*, 23(4):2124–2149.

Odachowski, K. and Grekow, J. (2012). Using bookmaker odds to predict the final result of football matches. In *Knowledge Engineering, Machine Learning and Lattice Computing with Applications*, pages 196–205. Springer.

Pollard, R. (1986). Home advantage in soccer: A retrospective analysis. *Journal of Sports Sciences*, 4(3):237–248.

Pong, T. K., Tseng, P., Ji, S., and Ye, J. (2010). Trace norm regularization: Reformulations, algorithms, and multi-task learning. *SIAM Journal on Optimization*, 20(6):3465–3489.

Resnick, S. I. (2013). *Extreme values, regular variation and point processes*. Springer.

Rue, H. and Salvesen, O. (2000). Prediction and retrospective analysis of soccer matches in a league. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 49(3):399–418.

Silver, N. (2014). Introducing nfl elo ratings. `https://fivethirtyeight.com/datalab/introducing-nfl-elo-ratings/`. Accessed: 2016-05-30.

Sinclair, J., Taylor, P., and Hobbs, S. (2013). Alpha level adjustments for multiple dependent variable analyses and their applicability–a review. *Int J Sports Sci Eng*, 7(1):17–20.

Skellam, J. G. (1945). The frequency distribution of the difference between two poisson variates belonging to different populations. *Journal of the Royal Statistical Society. Series A (General)*, 109(Pt 3):296–296.

So, A. M.-C. and Ye, Y. (2007). Theory of semidefinite programming for sensor network localization. *Mathematical Programming*, 109(2-3):367–384.

Srebro, N. (2004). *Learning with matrix factorizations*. PhD thesis, Citeseer.

Srebro, N., Rennie, J., and Jaakkola, T. S. (2004). Maximum-margin matrix factorization. In *Advances in neural information processing systems*, pages 1329–1336.

Srebro, N. and Shraibman, A. (2005). Rank, trace-norm and max-norm. In *International Conference on Computational Learning Theory*, pages 545–560. Springer.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 111–147.

Vandenberghe, L. and Boyd, S. (1996). Semidefinite programming. *SIAM review*, 38(1):49–95.

Vounou, M., Nichols, T. E., Montana, G., Initiative, A. D. N., et al. (2010). Discovering genetic associations with high-dimensional neuroimaging phenotypes: a sparse reduced-rank regression approach. *Neuroimage*, 53(3):1147–1159.