Mutual Information Scheduling for Ranking

Hamza Aftab
Princeton University
haftab@princeton.edu

Nevin Raj
Harvard University
nraj@college.harvard.edu

Paul Cuff
Princeton University
cuff@princeton.edu

Sanjeev Kulkarni
Princeton University
kulkarni@princeton.edu

***Abstract*—In this paper, an efficient and flexible scheduling algorithm for estimating the ranking of n objects through pairwise comparisons is proposed. By successively picking pairs for comparison such that the mutual information between the parameter(s) of interest and the outcome of the comparison is maximized, we quickly obtain a good approximation of these parameters using few comparisons. In simulation, this scheduling algorithm performs better than randomly selecting pairs for comparison or selecting pairs according to a natural heuristic previously proposed. However, a main advantage to this principled approach is that it appropriately adapts to a variety of models and parameters.**

***Index Terms*—Ranking, pair-wise comparison, mutual information, scheduling, importance sampling.**

I. INTRODUCTION

From Internet search engines to economics, sports, music, video games and voting, establishing ranks or orders among several different objects is a prevalent part of society and our daily lives. Some mathematical and technological examples of ranking include PageRank™ [1] for ranking web-pages, TrueSkill™ [2] for ranking Xbox players, and Elo's rating system for ranking chess players [3]. Quite often, the information available for computing a rank of different objects is limited, such as in sports, where the observation usually are either a win or loss (and in certain sports even a draw) of one team or player over another.

Consider establishing a ranking from pair-wise comparisons [4]. In some situations, we may be able to choose which information we want to obtain by selecting the next comparison based on past observations. For instance, a number of websites have been implemented over the years around the same basic design of gathering pair-wise comparisons for the sake of establishing a ranking. For instance, allourideas.org, facemash (by the founder of Facebook), and kittenwars.com to name a few. In these situations, it is possible to design the technology to collect information in a clever way, to make the most of each comparison request. Although not always applied to sports, a judicious selection of pairs, or scheduling, can reduce the number of comparisons needed to establish a reliable rank. This is the idea behind tournament brackets [5]. This can be crucial when the costs associated with each comparison are high, even justifying extra calculations needed for algorithmic scheduling.

The motivation for a better scheduling method comes from examining the simplest scheduling method, namely random selection. By picking every pair with equal probability, regardless of the strength of the two objects, we risk the chance of an unbalanced match-up and, hence, a predictable result. A more intuitive method suggests that the probability of selecting a pair should be proportional to how uncertain we are about their ranks and how evenly they are matched [6]. We suggest a systematic way to perform such an effective scheduling for a flexible set of ranking goals, which is simply based on the idea of maximizing mutual information.

II. PROBLEM STATEMENT

Consider a set of n objects $\{X_1, ..., X_n\}$ that are in competition with one another, such as sports teams, ideas, political candidates, search results, or investment portfolios. We refer to these objects as *players*. The players face each other in a sequence of pair-wise comparisons, or *matches*. The binary outcome of a match (denoted O_k for the kth match) has some randomness involved, so the same outcome may not result the next time the same two players are matched. Furthermore, the players' intrinsic skill levels, which influence the outcomes of the matches, are not equal. It is these skill levels that we are interested in estimating, based only on observing the outcomes of a sequence of matches. The main innovation we hope to contribute is a principled approach to scheduling future matches based on past outcomes.

Let the body of knowledge retained by the system after k matches be a function f_k of past outcomes:

$$\Omega_k \quad = \quad f_k(O_k, \Omega_{k-1}). \tag{1}$$

This information might contain the entire history of matches and outcomes or might contain some summary statistics. Regardless, we assume that we have a way of updating this state in real time as we get results from more comparisons.

The specific parameter of interest that we seek to estimate, R, is quite flexible. It can be any property related to the skills of the players, such as a sorted listing of all the objects in order of their average strength, the probability that a certain player wins over another in a match, or something simple like the "best" player among all n players.

After each match is observed, a new estimate $\hat{R}_k$ is made, defined by the *estimator* $L = \{L_k\}$:

$$\hat{R}_k \quad = \quad L_k\left(\Omega_k\right). \tag{2}$$

Also, the pair (i_{k+1}, j_{k+1}) that will be compared for the subsequent match is determined by the *scheduler* $S = \{S_k\}$:

$$(i_{k+1}, j_{k+1}) \quad = \quad S_k\left(\Omega_k\right). \tag{3}$$

The performance of a pair of estimators and schedulers will depend on the particular probabilistic model that is adopted, which relates the parameter of interest to the outcomes observed. We chose to use a linear model, in particular the Thurstone-Mosteller model [7] [8], for testing and simulations. In this model, each player X_i has a skill level which is simply a real number λ_i. Each outcome of a match is conditionally independent given the players that were involved, and the probability that player X_i beats player X_j is a function of the difference in skill, $\lambda_i - \lambda_j$ [9]. Therefore, these skill levels imply a ranking as well as all probabilities of outcomes of matches. This model is qualitatively very similar to the Bradley-Terry model [10] [11]. We expect that similar results will hold for a variety of models.

For more discussion of the model used, see Appendix VIII-A.

In real-world settings, the performance of an estimator and scheduler pair could be tested using real data and evaluated through cross-validation. In this work, we explore an idea for scheduling and test it against synthetic data. We generate data from the probabilistic model that we have assumed, with randomly selected parameters (skill levels). These parameters are not available to the estimator and scheduler, but they are used to measure the quality of the estimate.

III. Maximum Likelihood

Armed with a probabilistic model for the outcome of a match, based on the strengths of the players, we can employ any number of estimation techniques. The most common and conceptually simple technique is maximum likelihood, which involves finding the set of parameters which would maximize the probability of the outcomes that have been observed.

Many sports ranking algorithms use maximum likelihood to fit model parameters. However, some famous systems, such as Elo's chess ranking system, use other parameter update rules [3]. The updates in Elo's chess ranking algorithm can be understood as gradient ascent steps for maximum likelihood. This causes the ranking to be more affected by recent results than by older results.

We use maximum likelihood to estimate ranking, based on pairwise comparisons and the Thurstone-Mosteller model [12] [13]. The specific equation that is optimized is discussed in Appendix VIII-A.

IV. Mutual Information Based Scheduling

After observing some results of matches, it is possible to selectively schedule future matches to gain the most useful information for estimating a ranking. Thus, the goal of the scheduling algorithm is to obtain information about R quickly.

We suggest borrowing a quantity from information theory to quantify the amount of information that is obtained through a sequence of matches. Specifically, we suggest designing the matches to maximize the mutual information between the parameters of interest R and the outcomes $O_1, ..., O_m$.[1] We approximate such an optimization using a greedy maximization of $I(R; O_m | O_1, ..., O_{k-1})$ for each k.

Mutual information characterizes the average reduction in entropy of one random variable when another is observed. It can be defined a number of ways. In our case it is convenient to define it in terms of entropy. For an arbitrary collection of random variables X, Y, and Z, the mutual information $I(X; Y)$ is given by

$$\begin{aligned} I(X;Y) &= H(X) + H(Y) - H(X,Y) \\ &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X), \end{aligned}$$

and conditional mutual information $I(X; Y|Z)$ is given by

$$\begin{aligned} I(X;Y|Z) &= H(X|Z) + H(Y|Z) - H(X,Y|Z) \\ &= H(X|Z) - H(X|Y,Z) \\ &= H(Y|Z) - H(Y|X,Z), \end{aligned}$$

where the entropy (or differential entropy for continuous random variables) is given by

$$H(X) = \mathbf{E} \log \left(\frac{1}{p(X)} \right),$$

and the conditional entropy is given by

$$H(X|Y) = \mathbf{E} \log \left(\frac{1}{p(X|Y)} \right).$$

Here $p(x)$ represents the distribution of the random variable X, meaning if X is a discrete random variable then $p(x)$ is the probability mass function for X, and if X is a continuous random variable then $p(x)$ is the probability density function.

Let the *mutual information scheduler* be defined by

$$S_k(\Omega_k) = \arg\max_{(i_{k+1}, j_{k+1})} I(R; O_{k+1} | O_1, ..., O_k). \quad (4)$$

Intuitively, if we want to estimate R well, then we need the conditional entropy of R given the observations $O_1, O_2, ...$ to be small. Fano's inequality (Theorem 2.10.1 in [14]) provides a concrete connection between low probability of error in detecting R and low conditional entropy $H(R|O_1, ..., O_m)$. Since the prior entropy $H(R)$ does not depend on the scheduler, the chain rule for mutual information gives the following observation:

$$\begin{aligned} & \arg\min_{\{S_k\}} H(R|O_1, ..., O_m) \\ = \; & \arg\min_{\{S_k\}} \left(H(R) - I(R; O_1, ..., O_m) \right) \\ = \; & \arg\max_{\{S_k\}} I(R; O_1, ..., O_m) \\ = \; & \arg\max_{\{S_k\}} \sum_{k=1}^{n} I(R; O_k | O_1, ..., O_{k-1}). \quad (5) \end{aligned}$$

Thus, we see that the mutual information scheduler defined in (4) is a greedy approximation to maximizing the sum in (5). An alternate method would be to come up with a dynamic roadmap for the next few matches at a time. While the dynamic method that optimizes over several steps may have performance advantages, the greedy algorithm is significantly simpler computationally and performs well.

Note that the mutual information scheduler is defined generally. The definition does not specify a particular parameter R or a particular model. In fact, Hajek and Spuckler suggest using this principle for an entirely different problem of detecting connectivity of a graph by probing edges [15]. Thus, the concept of greedily maximizing mutual information is versatile. In our case, the benefit is that mutual information scheduling can be used regardless of what our ranking goals might be. The flexibility and accuracy of the idea are shown by the results in the next section, which indicate how well it performs in different scenarios.

Computationally, the suggested mutual information scheduler involves, at each point k, considering every possible pair of players, calculating the entropy of the outcome of a match between them conditioned on all of the past observations, and also calculating the entropy of the outcome conditional on R and the past observations (averaged over the posterior distribution of R), and selecting the pair that gives the greatest difference of the two. The entropy calculation is not tractable analytically, and numerically it involves a high dimensional integral.

[1] It should be noted that mutual information with respect to a parameter R is only well defined if R is indeed a random variable. Thus, we turn to a Bayesian framework, where we assume a prior distribution on R. In our case, we assume the skill levels in the Thurstone-Mosteller model are normally distributed.

We estimate the entropies through importance sampling as described in Appendix VIII-B and VIII-C.

Note that mutual information scheduling is not the same as simply finding the match for which you are least certain about the outcome. The entropy of the outcome is only one of the two terms in mutual information. In some cases they are the same. This occurs if the outcomes are not random (deterministic sort) or the outcomes are equally noisy regardless of known parameters (noisy sort). In those cases, mutual information scheduling is simply a matter of maximizing the entropy yielded by each match (i.e. choosing the most well-matched pair).

V. Results

We compare the performance of the proposed scheduling algorithm to two others: random scheduling and a heuristic found in [6] which suggests that we first pick a player that has not appeared in many matches and second, a player which is most equally matched to the first from the data available. The heuristic probabilistically selects the players according to the following rules:

$$P(\text{i as first player}) \propto \left(\frac{1}{\epsilon + n_i}\right)^{\alpha}$$

where n_i is the number of comparisons involving *i* and:

$$P(\text{j as second object}) \propto (min[\hat{p}_{i,j}, \hat{p}_{j,i}])^{\beta}$$

where $\hat{p}_{i,j}$ is our current belief about the probability of player *i* beating player *j*. Intuitively, both these criteria make sense for selecting a good match for the next comparison. Notice that this scheduling method involves some parameters which need to be tuned, which is one drawback to the algorithm. In order to give an honest comparison to this heuristic algorithm, we tuned the parameters so that the performance seemed to be at its best, minimizing the mean-squared error in the estimates of the skill levels.

We compare the three scheduling algorithms using simulated outcomes of matches, generated from the Thurstone-Mosteller model for pairwise comparisons. We also generate the skill levels according to a normal distribution (see Appendix VIII-A). All three ranking algorithms use maximum likelihood to fit the parameters (according to the Thurstone-Mosteller model). Also, mutual information is calculated with respect to this model.

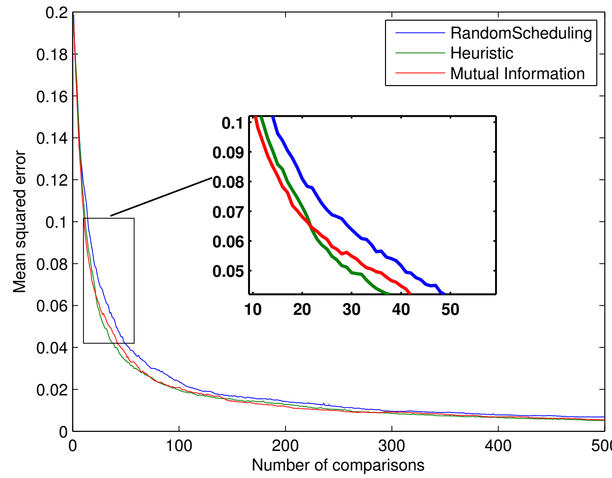

Figure 1. *Average mean squared-error between estimated and actual parameters.* The figure shows a different measurement of error to compare the scheduling methods and just like Figure 1, scheduling matches randomly does significantly worse than the others. However, the differences between the heuristic and the proposed algorithm after a number of probabilistic experiments are very small, with one performing better in some regions and vice versa. The mean skill levels are sampled from a normal prior distribution with standard deviation of 0.5, and the performances of the players are normally distributed around their mean skill's with standard deviation of 0.3.

A. Estimating skill levels

The first estimation problem that we consider is that of estimating the complete set of parameters in the model. For the Thurstone-Mosteller model (or any linear model, including Bradley-Terry and Condorcet), the parameters are the skill levels of the players. In figure 1, we compare the performance of the above mentioned scheduling algorithms, as measured by mean squared-error in the estimates of the skill levels.

Notice that mean squared-error, as a performance criterion for estimating model parameters, is quite model specific. We can also measure performance based on something that is model independent. In general, a model will specify the probability that an arbitrary player x beats another player y, for all pairs (x, y). This constitutes a matrix of winning-probabilities for each match. Usually, not all matrices of winning-probabilities are consistent with any one model. Thus, the Thurstone-Mosteller model, which we use, projects to a subset of winning-probability matrices consistent with the model.

Our second performance criterion used to compare the schedulers is measured directly from the winning-probability matrix, thus avoiding making a specific reference to the model parameters. We consider the difference between the estimated winning-probability matrix (based on estimated model parameters) and the true winning-probability matrix (based on true model parameters), and quantify this difference using the average Kullback-Leibler divergence between the true probability of winning and the estimated probability of winning, averaged over each pair.

Let the true winning-probabilities be $p_{i,j}$ and the estimated winning probabilities be $\hat{p}_{i,j}$. Then the average KL-divergence is given by

$$D(p, \hat{p}) = \frac{1}{n(n-1)} \sum_{i \neq j} p_{i,j} \log \frac{p_{i,j}}{\hat{p}_{i,j}}.$$

This measurement of error of the estimate, which we refer to simply as the *average KL-divergence*, has some nice motivation and properties. It is the answer to a very natural question. If a gambler were to bet on a sequence of independent matches between random pairs of players with some limited wealth that can compound if he bets wisely, and he placed bets optimally according to the estimated winning-probability matrix, how much will the error in his estimate hurt his winnings? The answer is that the exponent with which his wealth compounds will be reduced by the average KL-divergence.

Another nice property of the average KL-divergence is that it penalizes over-confident predictions. An estimated winning probability of 1% instead of a true winning probability of 10% will be penalized more than an estimate of 50% when the truth is 59%.

The performance of the three scheduling algorithms, as measured by average KL-divergence is shown in figure 2.

As Figures 1 and 2 show, with the mutual information maximizing algorithm, we learn faster in the beginning when we have to make the most out of fewer comparisons. The later differences become less prominent because all three rank estimators are consistent. However, it appears from Figures 1 and 3 that the ratio of the estimation errors

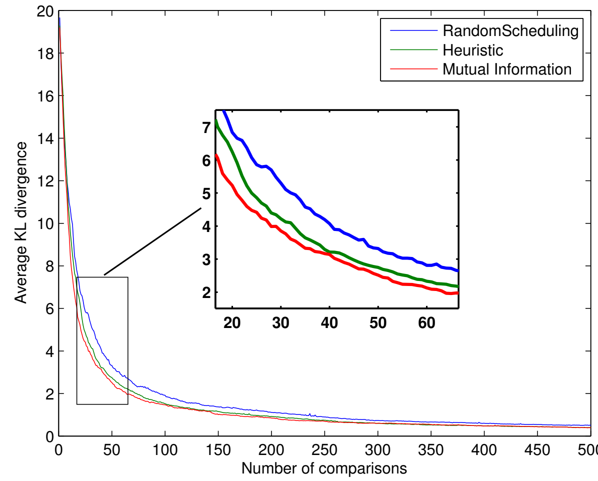

Figure 2. *Average KL-divergence between estimated and actual probability distributions.* The average KL-divergence (between actual and predicted winning probabilities) over a number of experiments is a measure of the error in prediction for the three scheduling methods. This measure of error declines most rapidly when scheduling is done through the proposed mutual information based algorithm, with the heuristic performing very close to but still distinguishably worse than it. The mean skill levels are sampled from a normal prior distribution with standard deviation of 0.5, and the performances of the players are normally distributed around their mean skill's with standard deviation of 0.3.

when scheduling via mutual information to the naive random scheduling approach a constant less than one.

B. Estimating a discrete ranking

The flexibility of the proposed algorithm means that it can be fine tuned for different goals. If we are interested only in deciding whether one player is better than another, rather than estimating how much better, we would calculate the mutual information with respect to the discrete order of the players instead.

One natural way to evaluate performance in rank estimation is to calculate the number of inversions in the rank: the number of consecutive swaps needed to make the estimated order agree with the actual one.

As Figure 3 shows, the improvement due to the suggested scheduling method is more pronounced. Note that we have also shown the performance when we calculate mutual information with respect to the winning probabilities. It performs worse than expected, since we are giving too large a weight

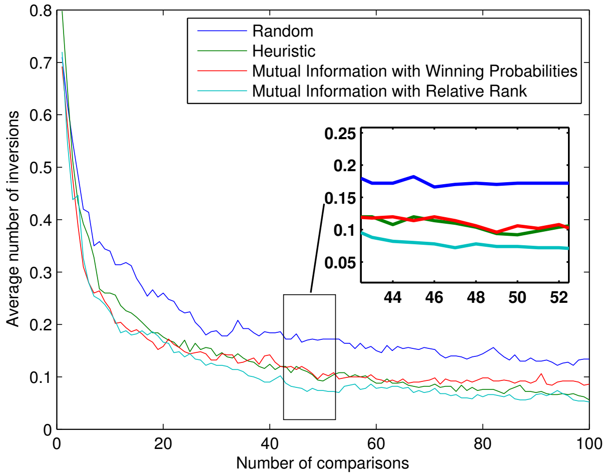

Figure 3. *Average number of inversions in the predicted rank order.* In order to quantitatively analyze the error in prediction of a discrete ranking, we found the number of swaps between consecutively ranked players to make the predicted order agree with the actual one. The suggested algorithm, based on maximizing the mutual information with the rank order, seems to give the lowest average number of inversions over the length of the experiments. The difference in error does not seem to be too significant, however these results show that the simple idea of maximizing mutual information seems to be a serious and consistent contender to the others. The mean skill levels are sampled from a normal prior distribution with standard deviation of 0.5, and the performances of the players are normally distributed around their mean skill's with standard deviation of 0.3.

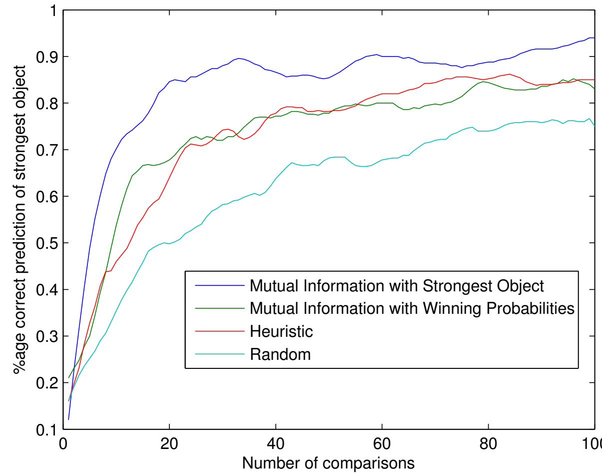

Figure 4. *Probability of correct prediction of the strongest player* The figure shows the fraction of times that we have a correct guess of who the strongest player is at different points in the experiments. As expected, the suggested scheduling method of maximizing the mutual information with the parameter that we wish to learn about (i.e who is the strongest player) performs significantly better than even the other mutual information based algorithm. The mean skill levels are sampled from a normal prior distribution with standard deviation of 0.5, and the performances of the players are normally distributed around their mean skill's with standard deviation of 0.3.

to pairs of objects whose strengths are relatively uncertain even though we know that one is better than the other.

C. Identifying the best player

As another example for which the scheduling method can be used, we consider identifying the strongest player in the group. Here we simply adjust the mutual information scheduling algorithm to calculate mutual information between the identity of the strongest player and the outcome of the potential next matches. This allows the algorithm to focus on finding only the best player. Figure 4 shows the proportion of times we pick a correct winner as a function of the number of comparisons.

As expected, there is a considerable improvement in results when we calculate mutual information with respect to the strongest player, even when compared against the other mutual information based scheduling method. The difference in performances of the algorithms is even more pronounced here than in the previous scenarios, since here we are only interested in a very narrow subset of information about the players, a fact that other algorithms ignore. However, this is not the only way to measure how good the predictions are. Figure 5 shows the performance of the four algorithms in terms of the minimum number of inversions required to bring the actual strongest object to the top of the predicted rank order. In this way, we consider not only whether the correct strongest object was predicted or not, but also how far off the prediction was. Once again, the results suggest the same, showing the recommended scheduling method considerably outperforming the others.

VI. Conclusion

This paper presents a robust and efficient method for dynamically matching objects for comparison, based on the mutual information that the comparison will provide. It is particularly useful for scenarios in which a rank must be established with only partial information. We have shown that the proposed method outperforms competing algorithms by obtaining better accuracy or at least offers serious competition for a wide variety of ranking scenarios.

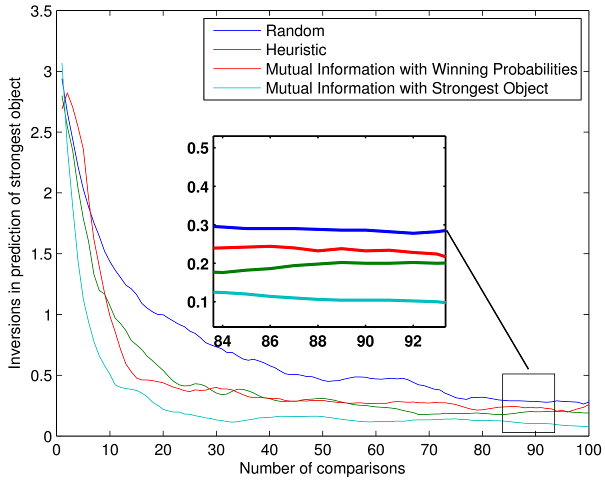

Figure 5. *Inversions in prediction of strongest object* The figure shows the distance of the actual strongest object from the top of the predicted rank order. The results are similar to those in figure 4, even though the error in prediction is now based on how wrong it is, rather than just it being right or not. Note that the error slowly reduces towards zero for all of the scheduling methods, however it seems like the ratio of the error when using the suggested method and the others remains a constant (not equal to one). The mean skill levels are sampled from a normal prior distribution with standard deviation of 0.5, and the performances of the players are normally distributed around their mean skill's with standard deviation of 0.3.

Particularly attractive about this algorithm is the fact that it is a methodological approach to solving the scheduling problem, and relies on a theoretically simple idea and not on scenario-specific details.

This work serves as a proof of concept for a mutual information-based scheduling algorithm, while ignoring the computational complexity of implementing such an algorithm and testing only against one simple model. While there may certainly be ways to improve this algorithm (less myopic optimization, for example), the exciting aspects of this algorithm are its simplicity, strong performance, and adaptability to various scenarios.

VII. Acknowledgment

Professor Adam Finkelstein regularly engaged in detailed conversations with the authors about this work and gave valuable insights.

VIII. Appendix

A. Thurstone-Mosteller model applied to ranking

Consider a system in which the performance of player i is normally distributed with unknown mean μ_i (i.e. the player's skill level) and known standard deviation σ [16]. In a match between i and j, the one with the higher performance wins. The probability of player i winning against player j is given by:

$$p_{i,j} = \frac{1}{2}\left(1 + erf\left(\frac{\mu_i - \mu_j}{2\sigma}\right)\right). \tag{6}$$

We assume that the skill levels $\mu_1, ..., \mu_n$ themselves have a normal prior distribution, with a known mean and standard deviation. The state of the system, Ω_k, consists of the $n \times n$ matrix of outcomes, B_k, such that $B_k(i,j)$ is the number of times player i has won against player j. Note that since every game results in either a win or a loss, $\Sigma_i \Sigma_j B_k(i,j) = k$. Any possible desired information about the rank can be extracted from the skill levels $\mu_1, ..., \mu_n$. We can estimate the skill levels given B_k using maximum likelihood estimation[2]:

$$\arg\max_{\{\mu_k\}} \left(\prod_k P\left(\mu_k\right)\right)\left(\prod_{i,j} {p_{i,j}}^{B_k(i,j)}\right), \tag{7}$$

where $P(\mu_k)$ is the probability of μ_i from the prior distribution. Note that this maximization can be reduced to a convex optimization problem by maximizing the logarithm of the probability.

B. Importance Sampling

Consider the vector of skill levels $\boldsymbol{\mu}$. The posterior distribution of $\boldsymbol{\mu}$ given the matrix of observed outcomes B_k is proportional to the joint distribution specified in (7). Specifically,

$$p(\boldsymbol{\mu}|O_1, ..., O_k) = \frac{p(\boldsymbol{\mu}, B_k)}{p(B_k)}, \tag{8}$$

where

$$p(\boldsymbol{\mu}, B) = \left(\prod_k P\left(\mu_k\right)\right)\left(\prod_{i,j} {p_{i,j}}^{B_k(i,j)}\right), \tag{9}$$

and $p(B_k)$, the probability of all the observed outcomes, is a normalization constant.

In order to perform mutual information scheduling, we need to calculate expected values with respect to $p(\boldsymbol{\mu}|B_k)$. We do this using importance sampling.

[2]Technically we are using maximum a posteriori estimation because we have assumed a Bayesian prior.

Suppose we have a distribution $q(\boldsymbol{\mu})$ from which we can easily draw samples. The expected value of a function of μ with respect to $p(\boldsymbol{\mu}|B_k)$ can be stated as

$$\mathbf{E}_{p(\boldsymbol{\mu}|B_k)}\left(f(\boldsymbol{\mu})\right) \;=\; \mathbf{E}_q\left(f(\boldsymbol{\mu})\frac{p(\boldsymbol{\mu}|B_k)}{q(\boldsymbol{\mu})}\right).$$

We can approximate the desired expected value by taking independent samples from $q(\boldsymbol{\mu})$ and calculating the sample mean. This is the idea behind importance sampling. The number of samples needed for a good estimate will depend on the choice of q.

After each new observation, we iteratively select a jointly Gaussian distribution q_k for importance sampling, based on the distribution q_{k-1} that was selected after the previous observation. Our method is as follows: We first estimating the quantity $p(B_k)$ needed to compute $p(\boldsymbol{\mu}|B_k)$, using importance sampling based on q_{k-1}. Then we estimate the conditional mean and covariance associated with $p(\boldsymbol{\mu}|B_k)$, again using importance sampling based on q_{k-1}. The resulting estimated mean and covariance are used for the jointly Gaussian distribution q_k. This approach gives a distribution $q_k(\boldsymbol{\mu})$ from which we can easily draw samples and that is closely matched to $p(\boldsymbol{\mu}|B_k)$. We can use q_k for importance sampling as we calculate other quantities such as entropy and mutual information, allowing us to get a good estimate quickly in a high dimensional space.

C. Estimating Mutual Information

The calculation of mutual information simplifies if we expand it the right way. Again, for brevity, we use B_k to represent the win-loss matrix of all observations up to match k.

$$I(R;O_k|B_{k-1}) \;=\; H(O_k|B_{k-1}) - H(O_k|R,B_{k-1}).$$

Since O_k is a binary random variable, we can explicitly state the above entropy expressions in terms of the binary entropy function $h:(0,1)\to(0,1)$ given by

$$h(x) \;=\; x\log\frac{1}{x} \;+\; (1-x)\log\frac{1}{1-x}.$$

Therefore, the mutual information to be maximized is

$$\begin{aligned} I(R;O_k|B_{k-1}) \;=\;& h(\mathbf{E}\left(O_k|B_{k-1}\right)) \\ & - \mathbf{E}\; h(\mathbf{E}(O_k|R,B_{k-1})). \end{aligned}$$

If the parameter of interest, R, is the complete set of parameters of the model, then the quantity $\mathbf{E}(O_k|R,B_{k-1}) = \mathbf{E}(O_k|R)$ comes directly from the model. In this case, calculating mutual information comes down to calculating an average inside the argument of h and another average of the output of h and taking the difference.

If R is not the complete set of parameter of the model, then it is a function of the parameters of the model. In this case, the method of importance sampling can be adjusted to calculate conditional expected values by categorizing the samples into appropriate bins.

REFERENCES

[1] T. H. Haveliwala. “Efficient Computation of PageRank.” Stanford University, 1999.

[2] P. Dangauthier, R. Herbrich, T. Minka and T. Graepel. “TrueSkill Through Time:Revisiting the History of Chess.” NIPS, 2007.

[3] A. P. Elo. “The Rating of Chessplayers, Past and Present.” Batsford, London, 1978.

[4] H. A. David. “The Method of Paired Comparisons.” Hafner Publishing Company, New York, 1963.

[5] H. Buhlmann and P. J. Huber. ”Pairwise Comparison and Ranking in Tournaments.” *The Annals of Mathematical Statistics*, Vol. 34, No. 2 (Jun., 1963), pp. 501-510.

[6] A. Tariq and A. Finkelstein. “Ranking Algorithms.” Princeton University, 2010.

[7] T. Leonard. “An Alternative Bayesian Approach to the Bradley-Terry Model for Paired Comparisons.” *Biometrics*, Vol. 33, No. 1 (Mar., 1977), pp. 121-132

[8] W. H. Batchelder and N. J. Bershad. “The Statistical Analysis of a Thurstonian Model for Rating Chess Players.” *Journal of Mathematical Psychology*, Vol. 19, pp. 39-60, 1979.

[9] L. L. Thurstone. “A Law of Comparative Judgement.” *Psychological Review*, Vol. 34, pp. 273-286, 1927.

[10] R. A. Bradley and M. E. Terry. “The Rank Analysis of Incomplete Block Designs. I. The Method of Paired Comparisons.” *Biometrika*, Vol. 39, pp. 324-345, 1952.

[11] R. A. Bradley. “Some Statistical Methods in Taste Testing and Quality Evaluation.” *Biometrics*, Vol. 9, pp. 22-38, 1953.

[12] R. B. Latta. ”Composition Rules for Probabilities from Paired Comparisons.” *The Annals of Statistics*, Vol. 7, No. 2 (Mar., 1979), pp. 349-371.

[13] S. I. Gass. ”Tournaments, Transitivity and Pairwise Comparison Matrices.” *The Journal of the Operational Research Society*, Vol. 49, No. 6 (Jun., 1998), pp. 616-624.

[14] T. M. Cover and J. A. Thomas. ”Elements of Information Theory”. Second Edition, Wiley-Interscience, 2006.

[15] B. Hajek and C. Spuckler. “A Martingale Framework for Trust.” *IEEE Information Theory Workshop*, Plenary Talk, Cairo, 2010.

[16] F. Mosteller, “Remarks on the Method of Paired Comparisons. I. The Least Squares Solution Assuming Equal Standard Deviations and Equal Correlations.” *Psychometrika*, Vol. 16, pp. 3-9, 1951.