

Estimating End-to-End Performance by Collaborative Prediction with Active Sampling

Irina Rish and Gerald Tesauro
IBM T.J. Watson Research
Hawthorne, NY 10532, USA
Email: {rish,gtesauro}@us.ibm.com

Abstract—Accurately estimating end-to-end performance in distributed systems is essential both for monitoring compliance with service-level agreements (SLAs) and for performance optimization (e.g., choosing the highest-bandwidth server for a download request in a content-distribution system). Due to infeasibility of exhaustive pairwise measurements, a natural alternative is to predict unobserved end-to-end performances from available historic data, with minimal additional measurements. In this paper we present an approach to this based on Collaborative Prediction (CP), an estimation method designed to work with sparse data, that has enjoyed much success in other domains (e.g. product recommendation systems), and obviates the need for *landmark nodes* commonly assumed in other approaches. Specifically, we use Max-Margin Matrix Factorization (MMMF), a linear factor model for CP that has outperformed state-of-art CP techniques. Moreover, our approach readily admits active sampling based on prediction confidence, and we further propose a novel active-sampling CP approach yielding even higher predictive accuracy, while allowing a flexible trade-off between “exploration” (choosing suboptimal samples to improve estimation accuracy) and “exploitation” (choosing node with best estimated performance). We demonstrate successful empirical results on a variety of practical problems, including network latency prediction (NLANR-AMP, P2PSim and PlanetLab datasets) and bandwidth prediction in content-distribution systems (IBM’s downloadGrid data).

I. INTRODUCTION

In any distributed system or network, accurate estimates of end-to-end performance between source and destination nodes (equivalently, between clients and servers) are vital to achieving the system’s Quality of Service (QoS) objectives for its customers. Typical QoS objectives relate to performance metrics such as round-trip time of HTTP requests, connection bandwidth, and network delays or latencies. In order to satisfy the QoS objectives, numerous routing or node choice decisions must be made, with the aim of selecting a satisfactory or optimal node to interact with from a large number of possibilities. Examples include choosing lowest-latency peers to communicate with in overlay networks, routing lookup requests in Distributed Hash Tables (DHT) to the nodes that are likely to respond fast, or choosing a high-bandwidth mirror site from which to download files in a content-distribution system (CDS). The difficulty of making such choices relates in part to the type of Service Level Agreement (SLA) established with customers of the system. A common industry practice for SLAs is to adjudicate transactions with respect to a binary performance threshold, and to stipulate penalties if a certain

percentage of transactions exceed the designated threshold. This type of SLA implies that management decisions need only find a “good enough” node choice with respect to the given threshold. However, with more general monotone objective functions (i.e., better performance is always more valuable), it could be worthwhile to extend the decision search to finding the globally optimal node over the entire network.

It is widely recognized that directly measuring end-to-end performance among all pairs of nodes is infeasible in large networks, and moreover, such measurements could not be kept up-to-date in highly dynamic environments. Hence in such networks, estimating end-to-end performance is naturally formulated as a problem of *inference*: given a relatively small set of pairwise end-to-end measurements, one needs to infer the likely performance between other pairs of nodes where there are no direct measurements. This understanding has motivated a variety of recent approaches proposed for predicting network delays/latencies. A common approach is to embed nodes in a low-dimensional Euclidean space using exhaustive measurements to a set of landmark nodes [?], [?], [?]. Another approach, called Vivaldi [?], seeks a minimum energy positioning by analogy with a network of physical springs. Finally, matrix-factorization approaches based on Singular Value Decomposition and Non-negative Matrix Factorization, were recently proposed by [?].

While such approaches have performed well in interesting scenarios, they face some potentially significant practical limitations. We note that the assumption of Euclidean distance properties (symmetry and triangle inequality) may often be violated in practice, as observed in various studies [?], [?], [?], [?]. While the matrix-factorization approaches proposed in [?] appear to overcome such limitations, they still rely on another strong assumption, shared by many current network distance prediction techniques: namely, it is assumed that for a given set of *landmark nodes*, all pairwise measurements among them and between the hosts and the landmark nodes are available. This assumption may not always be realistic, particularly for end-to-end measurements that are either costly or impossible to obtain on demand (e.g., forcing peers in a content-distribution system to upload or download files to all other nodes).

Herein, we propose a more broadly applicable machine-learning approach to end-to-end performance estimation by formulating it as a *collaborative prediction (CP)*, or *sparse*

matrix approximation problem. Namely, given a large and very sparsely sampled matrix (e.g., only a few percent of entries may be observed in some applications), CP aims to predict the unobserved entries from the observed samples, assuming the entries have dependencies across rows and across columns. Typical applications of CP include online recommendation systems that predict user preferences towards products (e.g., movies, books), based on previously obtained ratings from other users. In distributed systems management, we wish to predict the end-to-end performance, such as latency or bandwidth, based on a limited number of available measurements between some pairs of nodes. A potentially big advantage of CP is that it makes no assumptions about distance-metric properties or the existence of landmark nodes, nor does it require additional instrumentation or ability to perform specific measurements on demand. However, if given such ability, CP can be enhanced by active sampling (also known as *active learning*) approaches that select most-informative samples in order to best improve the model accuracy. Active learning is currently a hot research topic in the machine-learning community, as it was shown theoretically (under certain assumptions) to provide exponential savings in the number of samples required to attain a given accuracy [?], [?], and has also demonstrated practical success in many applications.

In this paper, we make use of the recently proposed *Maximum-Margin Matrix Factorization* (MMMF) [?] approach, which was shown to outperform state-of-art CP methods while also providing some nice theoretical guarantees. MMMF uses a convex-optimization formulation that bounds norm instead of rank, and is guaranteed to find a globally optimal solution, whereas non-convex, bounded-rank SVD-like methods could get stuck in bad local minima. MMMF is also more flexible than previously proposed SVD-like methods for network distance prediction since it can deal with arbitrary sparse matrices, instead of relying on a complete set of measurements associated with a fixed landmark nodes. Finally, MMMF can be viewed as simultaneous learning of a collection of *support vector machine* (SVM) classifiers, which have garnered a great deal of recent interest among machine learning researchers, due to their ability to consistently yield low prediction errors in many different kinds of applications, while preserving robustness to overfitting. The known properties of SVMs led us to hypothesize that they may perform well in end-to-end performance prediction applications, and in fact our empirical results demonstrate successful application of MMMF to prediction of end-to-end bandwidth and latency across a variety of network datasets.

Next, we propose a novel algorithm that further augments collaborative prediction approaches, and MMMF particularly, with active sampling and yields considerable improvements in accuracy for a given number of samples. This is important in cases where sampling has high cost, e.g., a user may become annoyed if she is asked to rate many products or a network may become congested if too many measurements are performed. We exploit the relation between MMMF and SVM classifiers to extend MMMF using margin-based active-

learning heuristics, where the margin is used to estimate informativeness of a candidate sample, as suggested in [?]. A similar approach can be applied to any CP method that estimates a prediction confidence (e.g., in probabilistic hidden factor models such as aspect model [?], MCVQ [?], and so on). Our active approach allows a flexible trade-off between the *exploration* goal of choosing a sample to improve the model accuracy in the future, and the *exploitation* goal of choosing the server with highest expected performance for a given service request.

In investigating the potential of our CP/active sampling approaches, we initially study the task of binary performance prediction, i.e., predicting whether the performance lies above or below a fixed threshold value. We adopt this focus primarily due to the previously mentioned common industry usage of binary performance objectives in commercial SLAs. However, we recognize that accurate real-valued performance prediction may also be of interest in many applications, and given sufficiently promising results in binary prediction tasks, we hope to extend our approach in this direction in future work. One encouraging point in this regard is that it appears relatively straightforward to reformulate binary MMMF so that it is capable of making real-valued predictions.

To summarize, this paper makes following contributions:

- We formulate end-to-end performance estimation as a collaborative prediction (CP) problem, hypothesizing that CP can be successful due to existing similarities among nodes, as well as common components affecting different end-to-end transactions. This formulation opens the door for technology transfer from the area of sparse matrix approximations, both linear matrix-factorization techniques, and various non-linear (e.g., probabilistic) methods from the CP literature [?], [?], [?].
- The new approach has greater flexibility and more general applicability than the state-of-art in network distance prediction as it does not require landmark nodes, can work with arbitrary sparse matrices, and makes no particular assumptions regarding low matrix rank or Euclidean distance properties (symmetry and triangle inequality).
- A novel active-learning CP approach is built on top of state-of-art MMMF method, which allows significant improvement in predictive accuracy that can be achieved with a limited number of additional observations; several heuristics are given that provide different exploration vs exploitation trade-offs in active sampling.
- Promising empirical results are presented on several datasets that include NLANR, P2P and PlanetLab network latency data from [?], as well as more recent PlanetLab data [?], and bandwidth data from IBM's downloadGrid (an internal content distribution system).

II. RELATED WORK

Various prior approaches to predicting network distances each make particular assumptions about the data. For example, Global Network Positioning system [?] assumes that the network nodes can be embedded into a low-dimensional

Euclidean space, with latency between nodes viewed as a distance function. This is not a very natural assumption (latencies do not necessarily obey triangle inequality), and the embedding algorithm is known to converge slowly and depend on the initial assignment. Another approach – the algorithm called Vivaldi [?] – makes an analogy with a physical network connected by springs, and tries to place the nodes so that the potential energy of the system is minimized (which is again a questionable assumption for modeling network latencies, bandwidth etc). Yet another type of approach uses so-called Lipschitz embeddings into an N -dimensional space where a node’s coordinates are given by its distances to a set of N landmark nodes, and hosts with similar distances to landmark nodes are close to each other in the space – again, a questionable assumption, since having similar latency from nodes A and B to node C does not necessarily imply low latency between A and B in an arbitrary network. The dimensionality of Lipschitz embeddings may be further reduced using Principal Components Analysis [?], [?].

Recently, matrix-factorization techniques based on Singular Value Decomposition (SVD) and Non-negative Matrix Factorization (NMF), were applied by [?] to network latency prediction. The approach, called *Internet Distance Estimation Service (IDES)*, uses a set of landmark nodes, assuming all pairwise distances between those nodes are known, as well as the distances from all host nodes to and from the landmark nodes. IDES applies SVD or NMF matrix factorization to the complete distance matrix over the landmark nodes, and then iteratively computes the host-to-host distances based on such factorization and the host-to-landmark and landmark-to-host distances. This approach overcomes some problems related to Euclidean distance assumptions and is quite accurate, but its reliance on landmark nodes limits applicability in cases when end-to-end performance measures are costly or impossible to obtain on demand.¹

There exists only a limited amount of previous work on active sampling for collaborative prediction, which includes a *value-of-information* approach of [?] and Bayesian model averaging method of [?]. Both approaches are based on probabilistic hidden-factor models and computationally expensive procedures for choosing next active sample that require minimization of expected cost (or uncertainty). On the contrary, our active sampling is quite simple and inexpensive as it only compares the margin values produced by MMMF. Another related work proposes an active-sampling method for low-rank matrix factorizations [?] that requires a small number of users to provide the ratings of ALL products – a clearly unrealistic assumption in any large enough, practical recommendation system.

The importance of combining network distance prediction with active measurements was also realized in recent work by [?] evaluating effects of network distance prediction on overlay routing and content-distribution systems performance.

¹For example, in peer-to-peer content distribution systems one can easily record bandwidths for actual file downloads, but cannot force every user to download a file from a given set of servers.

However, the active approach of [?] does not aim to improve the model, but rather to improve the immediate decision (e.g., directly measuring latency to 10 nodes predicted to be the lowest-latency ones). This approach corresponds to our “least-uncertain-positive” sampling heuristic (described later), which gives little improvement in predictive accuracy of the model, although it may provide an immediate good decision, if the current predictive accuracy is *already* good enough.

Another active sampling approach for network latency prediction based on Bayesian experiment design was recently proposed by [?]. This approach relies on knowing the routing matrix, which may be unavailable in various practical scenarios (e.g., due to noncooperative administrative domains or elements invisible to ping/traceroute), while our approach does not require such knowledge and is also more general than [?] since its hidden-factor discovery mechanism can be applied to arbitrary end-to-end transactions beyond just network latencies (e.g., to application layer). Similar arguments apply to the recently proposed *iPlane* approach [?] for composing predicted performance based on measured segments of Internet path.

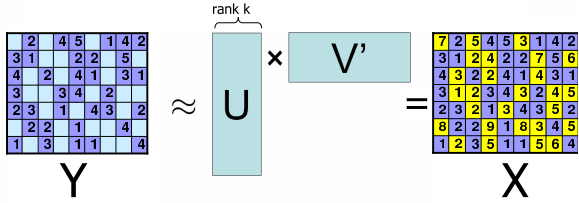
III. COLLABORATIVE PREDICTION AND MATRIX FACTORIZATIONS

Given a set of users and a set of products, the problem of collaborative prediction is to infer an unknown user’s preferences towards different products based on previously observed ratings for different (user, product) pairs. This problem originated in online product recommendation systems, but can also occur in networks, where there is a notion of a scalar preference metric between elements of the system.

For example, given a set of clients (peers that might request a service, such as file download), a set of servers (peers that can provide a service, e.g. have files of interest), and historic data measuring some performance metric for each client-server interaction (e.g., bandwidth), we wish to predict the performance of servers with respect to the given client and choose the best server. If we simplify this problem by using some aggregate metric for all interactions between a particular client-server pair, the data can be represented by a matrix where rows correspond to clients, columns correspond to servers, and the matrix entries represent a metric characterizing the (average) quality of a particular client-server interaction. (Note that the matrix can be extremely sparse: e.g., in some of our datasets, less than 1% of the matrix elements were filled). A similar approach can be used for predicting network delays between pairs of nodes given the delay information between some other pairs.

Formally, a *collaborative prediction (CP)* problem can be stated as a matrix completion problem - given a partially observed $n \times m$ matrix Y , find a matrix X (having same size as Y) that “best” approximates the unobserved entries of Y with respect to a particular *loss function* (e.g., sum-squared loss for real-valued matrices, misclassification error in case of discrete matrices, etc.). The main assumption in CP is that the matrix entries are not independent, i.e., there exists shared

Matrix Factorization



Objective: find a factorizable $X=UV'$ that approximates Y

$$X = \arg \min_{X'} \text{Loss}(X', Y)$$

and satisfies some "regularization" constraints (e.g. $\text{rank}(X) < k$)

Fig. 1. A matrix-factorization approach to approximating a partially observed matrix.

properties across users and across products, which can be used to generalize from observed to unobserved entries.

A typical assumption underlying various CP techniques is a *factorial* model in which there are some unobserved "hidden factors" pertaining to users or to products that would affect a particular user's preference toward a particular product. For example, the genre of a movie, or its comedic or offensive language content, may systematically affect whether certain groups of users prefer it. Similarly, two nearby nodes within a large network may share several hidden factors, such as common intermediate nodes on their paths to a third node. There may also be other essentially distance-independent hidden factors, such as machine type and connection type, that may influence a node's QoS regardless of distance to other nodes. For example, a powerful server with a T3 internet connection may be able to consistently deliver high-bandwidth downloads even to very distant clients.

A particular example of factorial models are *linear factor models* where each factor is a preference vector, and actual user's preferences correspond to a weighted linear combination of these factor vectors with user-specific weights. Linear factor models yield a *matrix-factorization* approach that was successfully applied to various prior CP problems. We also expect that linear factor models may be particularly well-suited for domains such as latency prediction where the end-to-end measurements are additive, i.e., can be decomposed into a (weighted) sum of intermediate delays.

Formally, let k be the number of such factors, then the matrix Y can be approximated by a matrix factorization $X = UV$, where U is a $n \times k$ *coefficient matrix* (where each row represents the extent to which each factor is used) and V is a $k \times m$ *factor matrix* where the rows represent the "expression level" of the factors in each of m "products" (see Figure ??)². Since the rank of the approximation matrix X is clearly bounded by k , fixing k to some small value leads to a low-rank matrix factorization approaches.

For example, a standard matrix-factorization approach is singular value decomposition (SVD) which finds a low-rank

approximation that minimizes the sum-squared distance between X and a *fully observed* Y . The problem is, when Y is not fully observed, as in CP and particularly in end-to-end performance prediction, SVD is not directly applicable and finding a low-rank approximation to a partially observed function using a sum-squared loss becomes a difficult non-convex optimization problem, for which no exact solution method is known. Also, even for a completely known matrix Y , approximating it with respect to other losses than sum-squared loss (e.g., expected classification error) is still a non-convex optimization problem [?] with multiple local minima, which is hard to optimize since gradient methods typically used for such problems tend to get stuck in local minima, without any guarantees on the solution quality.

IV. MAXIMUM MARGIN MATRIX FACTORIZATION

Recently, a novel matrix factorization approach was proposed in order to overcome the limitations of the previously used non-convex approaches. This approach, called *Maximum Margin Matrix Factorization (MMMF)* [?], replaces the bounded-rank constraint of low-rank approximations by a *bounded norm* constraint on U and V , which in its turn corresponds to bounding the *trace-norm* (the sum of singular values) $\|X\|_{\Sigma}$ of X . Since the trace-norm is a convex function [?], minimizing it together with any convex loss function or constraint results into a convex problem which can be solved exactly by standard optimization techniques. Namely, for binary-valued matrices $Y \in \{-1, 1\}^{n \times m}$ (e.g., -1 and 1 denoting "bad" and "good" performance, respectively), MMMF uses the *hinge loss* objective as in max-margin linear discriminant (SVM) learning and results into the following optimization problem:

$$\min_X \|X\|_{\Sigma} + c \sum_{ij \in S} h(Y_{ij} X_{ij}), \quad (1)$$

where $\|X\|_{\Sigma}$ is the trace-norm of matrix X , c is a trade-off constant, S is the set of all non-missing entries in the given matrix Y , and $h(z) = \max(0, 1 - z)$ is the *hinge-loss*, a convex upper bound on the 0-1 loss, or misclassification error. (While minimizing classification error is the ultimate objective, it is difficult to optimize the 0-1 loss directly due to its non-convexity; thus, a popular approach in current machine-learning research is to replace 0-1 loss with various convex upper bounds such as the hinge loss mentioned above). The optimization problem formulated above is convex and can be now solved by standard semi-definite programming (SDP) solvers.

Note that MMMF has a very intuitive interpretation that will be later exploited by our active sampling approach: it can be also viewed as a simultaneous learning of feature vectors and linear classifiers (see Figure ??). Namely, assume a factorization $X = UV$ is found, the rows of the $n \times k$ matrix U can be viewed as a set of n *feature vectors*, while the columns of V can be viewed as *linear classifiers*, and the entries of the matrix X are the results of classification using these classifiers. The original entries in the matrix Y

²The explanatory figures ??-?? are augmented versions of the original slides used in the presentation by N. Srebro, obtained at <http://people.csail.mit.edu/nati/mmmf/>.

MMMF Idea

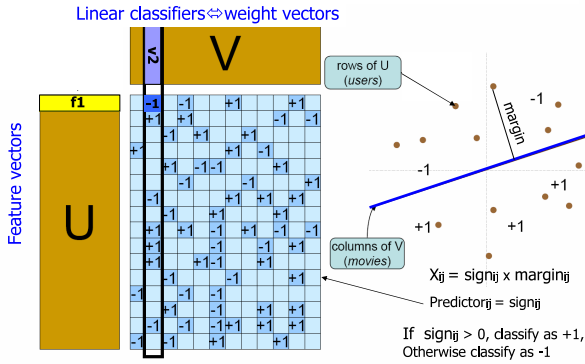


Fig. 2. MMMF idea: matrix factorization as a simultaneous search for (bounded-norm) feature vectors corresponding to rows and maximum-margin linear classifiers corresponding to columns. Margin usually refers to the minimal distance of a sample to the hyperplane dividing positive and negative examples; however, we will also use “margin” as a “distance” from a sample to the hyperplane.

can be viewed as *labels* for the corresponding feature vectors, and the task is to learn simultaneously a collection of feature vectors (rows in U) and a set of linear classifiers (columns in V) from a set of labeled samples (columns in the original matrix Y), such that a good prediction of unobserved entries can be made. A state-of-art in learning linear classifiers is the Support Vector Machines (SVM) approach, that attempts to find a linear separator between the examples of two different classes that maximizes the *margin* (the distance between the line and the closest example). SVM approach has theoretical guarantees regarding the generalization error, and typically perform very well in practice, which explains its popularity in the machine-learning community. There are also extensions of SVMs to multi-category and real-valued regression problems, but they are outside of scope of this paper; herein, we will focus on basic binary classification problem motivated by SLA threshold-based requirements.

In summary, the advantages of MMMF over the prior state-of-art in matrix factorization include a convex optimization problem formulation that allows to find the optimal solution instead of getting stuck in a local optimum, empirical performance reported in [?] that improves over the state-of-art collaborative prediction approaches, and a convenient interpretation that relates it to learning SVM classifiers.

V. ACTIVE SAMPLING WITH CP

Previously considered “passive” learning approaches assumed no control over the data collection process. However, in many applications, we have a choice between different actions that result in a new measurement. For example, in Internet distance prediction, we can decide to measure a distance between a particular pair of nodes; in content distribution systems, a particular mirror site needs to be selected to satisfy a file request which also leads to an additional bandwidth measurement; in an online recommendation system, we can choose a product to suggest to the current user, and so on. Such

Active MMMF Idea

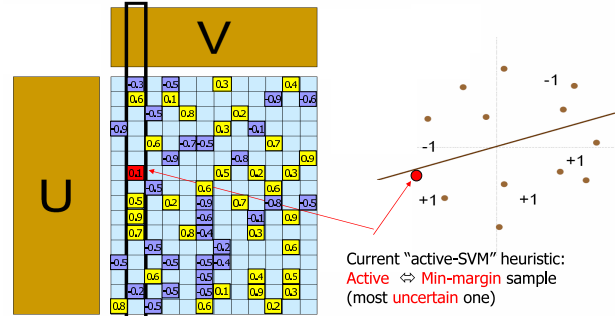


Fig. 3. Main idea of active learning in MMMF: choose the most “uncertain” sample next, where the margin (distance to linear classifier) measures the confidence in the prediction (i.e. we are least confident in predictions made for the instances closest to separating line between positive and negative examples). Note that the matrix is real-valued, with X_{ij} =distance (with sign) between feature-vector i and linear classifier j .

additional measurements can greatly improve the predictive accuracy of our model, but they also have a cost (e.g., potentially low bandwidth or high network latency if a poor server is selected). On one hand, we wish to choose the next sample which is most-informative and leads to greatest improvement in future predictive accuracy (i.e., yields better exploration), while on the other hand we want to avoid choosing samples which might be too costly by exploiting our current predictions about the sample costs (i.e., the corresponding predicted performance). Such exploration vs exploitation trade-offs must be considered as a part of our decision-making.

The interpretation of MMMF approach through learning a collection of SVMs provides a natural way of introducing a cost-efficient *active sampling* (or *active learning*). If our main concern is better exploration, i.e. choosing most-informative sample next, we can easily incorporate active SVM learning heuristics into MMMF. One example is a simple heuristic proposed by Tong and Koller [?] that prefers the *minimum-margin* sample (i.e. the one we are least confident about). For example, active sampling in a content distribution environment can be done either when a user requests a file and a centralized management system has to decide which mirror to assign to it, or when there is no particular file request but system decides to perform a test transaction between pair of nodes (clearly, such “enforced” downloads are only possible in centrally managed file distribution systems such as IBM’s downloadGrid). Empirically, such sampling tends to yield a better model than random sampling, as we demonstrate in the empirical section. The idea of min-margin active sampling is demonstrated in Figure ??.

To provide benchmark comparisons for our proposed *most-uncertain* sampling strategy, we also empirically investigate two alternative cost-insensitive strategies: *least-uncertain* (max-margin) chooses samples with highest confidence in

Active Max-Margin Matrix Factorization (A-MMMF)

Input: Sparse binary (-1/1) matrix \mathbf{Y} , batch size k , max # of new samples N , active-sampling heuristic h (most-uncertain etc).

Output: Full binary-valued matrix \mathbf{X} predicting unobserved entries of \mathbf{Y} .

Initialize: $Y' = Y$ /* currently observed data */

$N' = 0$ /* current number of active samples */

1. $\mathbf{X}' = \text{MMMF}(Y')$ /* compute full real-valued matrix \mathbf{X}' , where $|X'_{ij}| = \text{distance}(\text{feature-vector } i, \text{linear classifier } j)$, $\text{sign}(X'_{ij})$ predicts unseen Y_{ij} . */
2. $U = \text{set of unobserved entries of } Y'$
3. $S = \text{active_select}(h, X', U, k)$ /* select k best unobserved samples from U using heuristic h and current predictions X' */
4. Request labels for new samples $s_{ij} \in S$, and add them to Y' .
5. If $N' + k < N$
 $N' = N' + k$; goto 1
else return $\text{sign}(X')$ /* return only binary -1/1 predictions */

Fig. 4. Active max-margin matrix factorization (A-MMMF) algorithm.

prediction accuracy, as well as a baseline *random* sampling strategy. Besides these “aggressive” strategies which ignore any possible sampling costs, we have also studied other active sampling approaches that take into account the cost of sampling and may decide to be more “conservative” about sample choice, e.g., when sampling also means providing a service such as file download, where besides improving the future accuracy we are also concerned with the immediate cost of sampling. Within our binary prediction framework we assume that positive samples (e.g., high bandwidth or low latency) have less cost than negative samples. On this basis we additionally explore two cost-sensitive active learning heuristics: a *most-uncertain-positive* heuristic that chooses the min-margin sample from among samples currently predicted to be positive, as well as a *least-uncertain-positive* heuristic, which corresponds to a purely “greedy” strategy of trying to choose the sample with least expected cost.

Our active sampling algorithm (Active MMMF, or A-MMMF) is presented in figure ???. The algorithm assumes a particular active learning heuristic specified as an input.

The above five strategies lead to a number of interesting trade-offs regarding model accuracy and sampling cost, as we see empirically in the following section. Based on current active learning theory, we generally expect that the most-uncertain strategies should lead to learning the most accurate models for a given number of samples, and that least-uncertain sampling (training on data that the model is already highly confident about) should yield quite poor models. Beyond that, we are on less firm footing regarding any expected accuracy vs. cost trade-offs, as theories of “exploration vs. exploitation” are less well-developed and there are few if any rigorous results or definitive insights.

VI. EXPERIMENTAL EVALUATION

We have tested both simple MMMF as well as active sampling (A-MMMF) on a variety of datasets containing

end-to-end performance data. These include several publicly available network latency datasets that we borrowed from [?], and proprietary data from an IBM-internal file distribution system.

A. Data Sets

PL-RTT2003 dataset (from [?]) was originally obtained from PlanetLab pairwise ping round-trip time (RTT) measurement project [?]. A subset of *minimum* round-trip times measured at 3/23/2004 0:00 EST was selected, and missing values were filtered out, since none of the algorithms used in [?] could handle missing values. Namely, if there were some missing values either in a column or a row corresponding to some node, both the row and the column corresponding to such node were eliminated from the matrix [?].

PL2005 dataset was obtained directly from the PlanetLab project, and *average* round-trip times measured at 02/01/2005 0:00 were selected. We eliminated only rows or columns that corresponded to completely missing measurements (e.g., node serving only as a source or only as a destination), but unlike [?] we did not eliminate the corresponding node from the matrix, so the resulting matrices are not necessarily square.

NLANR-AMP dataset was obtained by [?] from the NLANR Active Measurement Project [?], that collects measurements between the pairs of nodes at NSF supported HPC sites, with about 10% of the nodes located outside of the US. All-pairs measurements were collected over 110 nodes on 01/30/2003; each host was pinged once per minute, and the minimum response time per day was chosen for each pair of nodes.

P2PSim dataset was obtained from the P2PSim project [?] that measured network latency among 2000 Internet DNS servers using King method [?] (the servers were taken from the Internet-scale Gnutella network trace).

dGrid2005 dataset was collected from *downloadGrid*, an IBM-internal file distribution system; the data were collected in Dec. 2005 over 10913 clients and 2746 servers, and contain the history of file downloads. The architecture of downloadGrid is similar in some respects to the Internet-based Gnutella, Napster and BitTorrent protocols as it allows peer-to-peer file downloading. However, it differs in utilizing a centralized decision-making architecture for matching “servers” (i.e. sources for downloadable files) with “clients” (i.e. download destinations). While the use of centralized decision-making is motivated primarily by security issues, the concomitant centralized data collection provides an opportunity for optimization of global system performance. By running algorithms such as MMMF on the aggregate system data, it may be possible to make reasonably accurate performance predictions for unobserved client-server pairs, and thereby make better assignments than could be made based solely on directly observed performance data.

From the history of file downloads in the downloadGrid, we created a matrix where each entry corresponds to the average bandwidth for a given (client, server) pair. The original matrix was extremely sparse (only less than 0.3 % of the entries were

observed) in that recorded performance data for any given node contains interactions with only a few other nodes. While MMMF has no trouble training on such data, for evaluation purposes only (i.e., in order to have enough data for testing the learned model), we have selected a 70x70 subset of the clients and servers that yield rows and columns most densely populated with recorded performance data. Currently, we also imposed same 70x70 size restrictions on the other datasets.

All of the real-valued performance measurements in each data set were transformed to a -1/1 binary representation by comparing the measured performance with a given threshold, typically chosen to lie at a certain percentage level (50%, 70%, or 90%) of the entire set of performance statistics in the particular dataset (i.e. 50% corresponds to median performance).

B. Basic MMMF results

In our initial experiments, we directly compared basic MMMF (i.e., not augmented by active sampling) with the IDES SVD and NMF algorithms described in [?]. Our evaluation methodology is as follows. We perform 10 trials for each 70x70 dataset, where in each trial we randomly select a subset of 20 “landmark nodes” with complete performance data between all pairs of landmarks. Then using the 20x20 landmark-landmark data, as well as 20x50 and 50x20 landmark-nonlandmark data (some entries of which may not be filled in) as training data, we train predictors using IDES-SVD, IDES-NMF and MMMF and evaluate their prediction accuracy on the set of 50x50 nonlandmark-nonlandmark connections. This methodology applies to all datasets except the downloadGrid dataset: due to its sparsity, we could not select a reasonably large subset of landmark with complete set of measurements to and from the remaining nodes, and thus we could not run IDES on this dataset. Instead, we tested MMMF only, by randomly splitting entries in the 70x70 matrix into 50% training and 50% test datasets.

Over our several datasets, our results generally show that MMMF is competitive with the IDES methods in datasets that conform well to the assumptions underlying IDES. Furthermore, MMMF can obtain a significant advantage over IDES in more general kinds of scenarios where the dataset characteristics deviate substantially from the IDES assumptions.

A good representative illustration of our results is shown below in Figures ?? and ?. In the results for the PL-RTT2003, NLANR-AMP and P2PSim datasets shown in Figures ??(a)-(c), we observe that the prediction accuracies of IDES-SVD, IDES-NMF and MMMF are fairly comparable at each of the five binary performance thresholds (0.5, 0.6, 0.7, 0.8 0.9), and there is no clear dominance among the algorithms. (Note that the error bars are quite large so the differences are not statistically significant.) On the other hand, a much more significant and interesting difference in prediction accuracy is found in the PL2005 dataset results shown in Figure ??(d). We see in this case that the prediction errors of the IDES algorithms are worse by a factor of 3-7 than the MMMF error, which remains below 10% for each threshold value. A possible

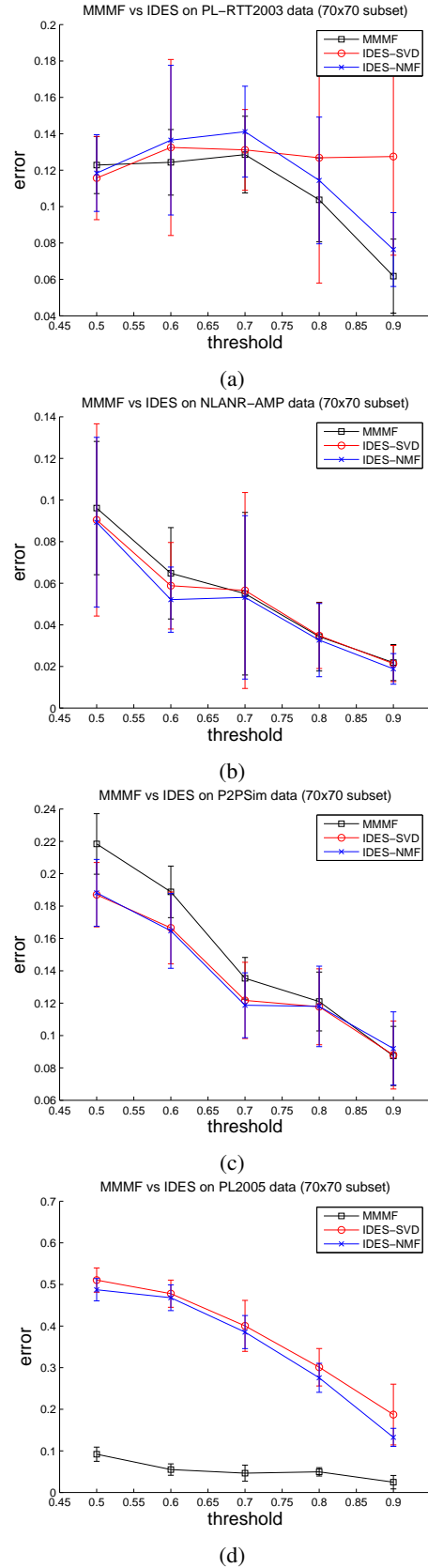


Fig. 5. Collaborative Prediction (MMMF) versus IDES methods (SVD and NMF) on 70x70 subsets of (a) PL-RTT2003, (b) PL2005, (c) NLANR-AMP and (d) P2PSim datasets.

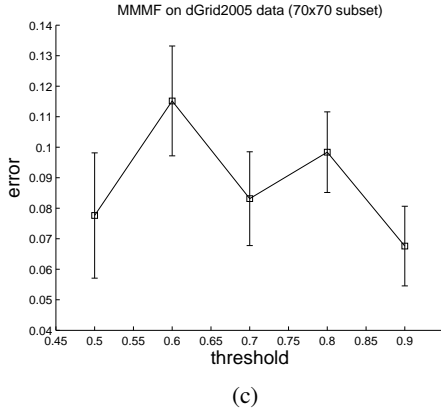


Fig. 6. Collaborative Prediction (MMMf) on the dGrid2005 dataset, MMMf only (IDES was not applicable).

explanation lies in the way the PL2005 dataset was processed: we removed individual empty rows or columns in the original 70x70 matrix (corresponding to nodes that never served as sources or as destinations, respectively) but in contrast to the processing of the PL-RTT2003 dataset in [?], we did not make the resulting matrix square by completely eliminating such nodes. The resulting rectangular matrix is now far from having distance matrix-like properties in two ways: it does not contain all pairwise distances (since it is not square), and perhaps more importantly, the diagonal elements deviate substantially from self-distances, which should all be zero. We hypothesize that such deviations from distance matrix properties account for the poor performance of the IDES methods, whereas as expected, MMMf was unaffected, since it can work in principle with arbitrary matrix data.

Finally, Figure ?? illustrates a scenario where there are no identifiable landmark nodes and thus IDES methods are completely inapplicable, yet MMMf easily accommodates this scenario, and in fact gives quite good prediction accuracy (less than 13% prediction error in all cases).

C. Active Learning Results

We now present our results applying A-MMMf (MMMf augmented by active sampling) to the same datasets described above. Our evaluation methodology is as follows. For each 70x70 dataset, we perform 20 independent trials, where in each trial we initialize the MMMf predictor by training on a randomly selected 5% of the matrix elements. We then randomly select another 50% of the matrix elements to be held out as test data. The remaining 45% of matrix elements serves as a source of samples that may be selected by various “active” heuristics. We progressively retrain MMMf in stages by selecting a batch of 50 samples from the active set, transferring them to the training set, and then recomputing the MMMf solution. We compare five different active selection heuristics here: “Most-uncertain” chooses samples that are closest to the margin of the current MMMf predictor. “Most-uncertain-positive” also uses the min-margin idea but further constrains the selected samples to be estimated positive (i.e. above threshold) by the current MMMf predictor. “Least-

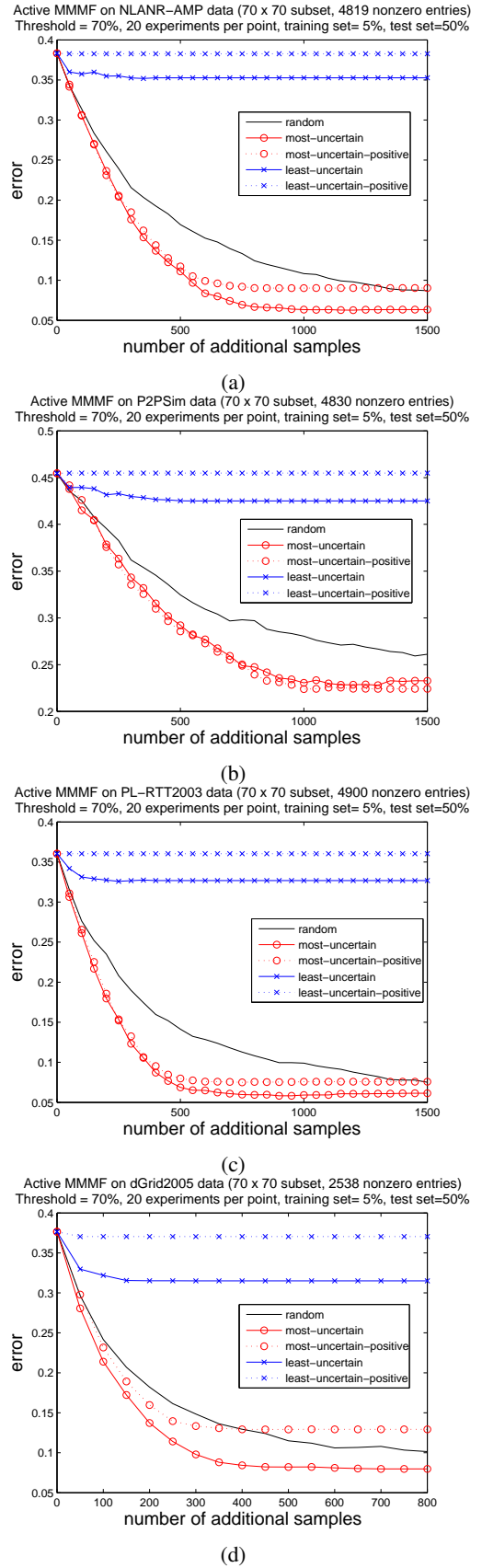


Fig. 7. Active learning results on (a) NLANR-AMP, (b) P2PSim, (c) PL-RTT2003 and (d) dGrid2005 datasets: improvement in the prediction error with the increasing number of samples.

uncertain” and “Least-uncertain-positive” apply the opposite principle of choosing samples that are furthest from the margin, i.e., samples for which MMMF is most confident in its prediction accuracy. Finally, “random” denotes the baseline uniform random sampling strategy.

Our results, plotted as mean test-set prediction error vs. number of additional samples selected, are shown in Figures ?? (We did not plot the error-bars here to avoid the clutter, but the differences were statistically significant.) The qualitative behavior seen in each dataset is highly consistent. In each case, we observe as expected that both of the “most-uncertain” strategies reduce prediction error more rapidly than with random sampling, and that both of the “least-uncertain” strategies provide quite poor choices of training samples, leading to very little improvement in prediction accuracy. Additionally, we note that, in accordance with active learning theory, the number of samples needed to reach a desired accuracy level may be significantly reduced when using the “most-uncertain” heuristics compared to using random sampling. For example, if we wish to reach the apparent asymptotic performance levels of the “most-uncertain” strategies, we would need only ~ 500 active samples in the PL-RTT2003 dataset, ~ 700 samples in the NLANR-AMP dataset, and ~ 1000 samples in the P2Psim dataset, vs. more than 1500 randomly chosen samples in each of these cases. A final point of interest is that the accuracy of the “most-uncertain-positive” strategy generally lies close to that of the “most-uncertain” strategy, and thus may provide a somewhat safer alternative in scenarios where choosing a sample that turns out to be negative (below threshold) entails a tangible cost of delivering poor performance to a customer.

In addition to minimizing the number of samples needed to train an effective CP module, it is perhaps of more salient interest to minimize whatever costs may be associated with acquiring the training samples. We have also examined this issue within our binary performance model by formulating a sampling cost model that is dominated by the SLA cost of poor performance. In this model, whenever a sampling strategy chooses a “negative” sample with below-threshold performance, we assign it a unit cost, whereas selecting a “positive” sample with above-threshold performance incurs no cost. In this way we can measure for each of our datasets the total sampling cost needed to reach a given prediction accuracy level, using each of our five candidate sampling strategies. Our results for four of our datasets are plotted below in Figure ??. We note in each case that plots of prediction error vs. cumulative cost are qualitatively similar to the corresponding plots of prediction error vs. number of training samples. Perhaps not surprisingly, this suggests a fairly linear relationship between the number of training samples and the total sampling cost, which would occur if negative samples are acquired at a fairly constant rate. We can see that for each dataset, usage of our min-margin sampling heuristics can yield very significant savings in total cost relative to random sampling, if one is interested in reaching low prediction errors where the curves flatten out. In the P2Psim dataset, we can obtain a prediction

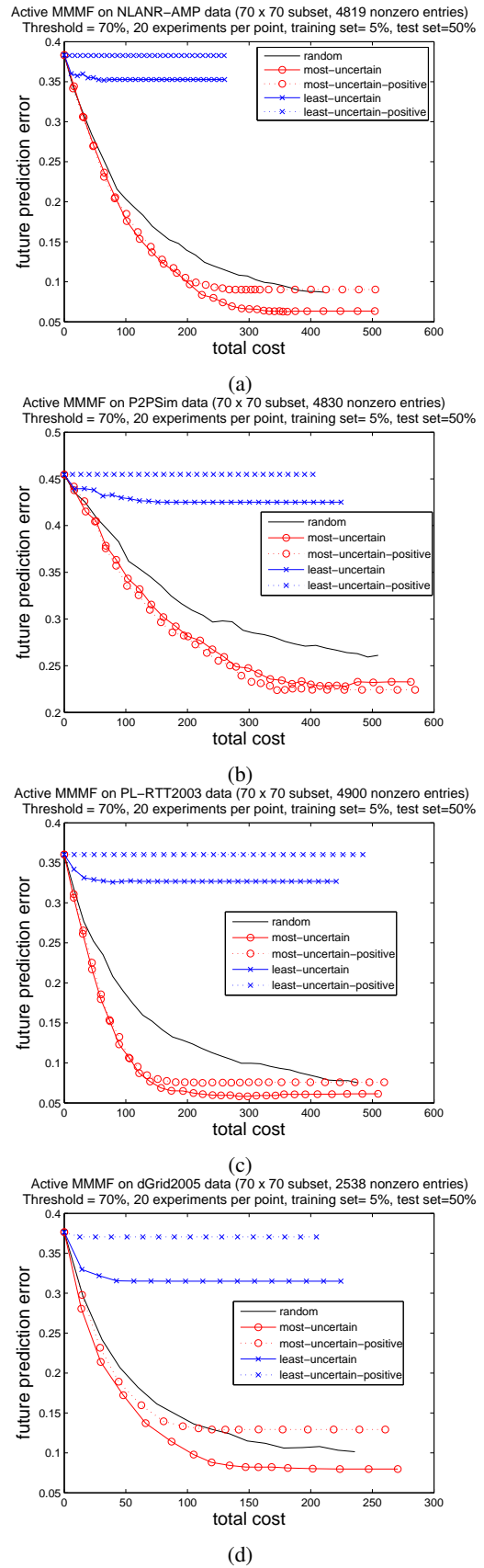


Fig. 8. Trade-off between the error reduction and the cost of active sampling on the (a) NLANR-AMP, (b) P2Psim, (c) PL-RTT2003 and (d) dGrid2005 datasets.

error of 0.25 at cost of ~ 270 , whereas with random sampling the cost would be over 500. In the NLANR-AMP dataset, we can reach 0.1 prediction error at a cost of ~ 200 using min-margin sampling, vs. a random sampling cost of ~ 350 . In the PL-RTT2003 dataset, with min-margin sampling we reach 0.1 prediction error at a cost of ~ 100 , vs. a cost of over 300 using random sampling. Finally, in the dGrid2005 dataset, min-margin sampling achieves 0.1 prediction error at a sampling cost of ~ 100 , compared to a random sampling cost of ~ 200 .

VII. CONCLUSIONS

We presented a new approach to end-to-end performance prediction that adopts and builds upon two important areas of current machine learning research, Collaborative Prediction (CP) and Active Learning. The CP framework, which was originally designed to learn to make inferences across users and across products based solely on sparsely sampled user-product interactions, can be effectively extended to pairwise interactions in networks (e.g. source/destination or client/server) as we saw in our empirical results. Many sophisticated learning algorithms developed for CP could prove to be useful in network settings, including MMMF, which delivered competitive performance in all of our datasets, and is free of limiting assumptions that restrict applicability of other end-to-end prediction techniques. The practicality of MMMF is further enhanced by adoption of Active Learning techniques that intelligently choose the best training samples for purposes of improving classifier accuracy. It is shown theoretically that under certain simplifying conditions active learning can provide exponential savings in the number of samples required to attain a given accuracy. Although no such theoretical guarantees are yet available for the active collaborative prediction method proposed in this paper, our empirical findings conform with this understanding, as we saw very substantial reductions both in the number of samples needed to reach a given performance level, and in the total cost of acquiring such samples.

Our greatest interest in future work is to extend our framework to encompass the dynamic aspects of both end-to-end performance prediction, as well as network management decisions based upon such predictions. Our current formulation ignores the dynamically changing nature of network states, which may render older measurement information obsolete, and the impact of decisions on states, which may necessitate a change in decision-making strategies. For example, if too many clients are directed to download a file from a “good” server, it could cause the server to become a “bad” server due to overloading. One approach we are investigating for dynamic inference is to extend MMMF by adding mechanisms for inference over time based on time-series analysis techniques. This could allow development of models of the decaying influence of older measurements on the current inference matrix, as well as forecasting methods predicting likely future states of the inference matrix based on how it has evolved in the past. The other major extension we are investigating

is combining dynamic inference models with models for sequential decision making, e.g., Markov Decision Process models. We are especially interested in Reinforcement Learning approaches to this, which would allow automatic learning of effective management policies without needing explicit models of management actions influence state transitions in the network.

ACKNOWLEDGEMENTS

We would like to thank Alina Beygelzimer, Rajarshi Das, and Jeff Kephart for many insightful discussions that contributed to the ideas of this paper. We also thank Nati Srebro for providing his MMMF code, and Yun Mao for providing IDES code and datasets.