, ,

Efficient Distributed Monitoring with Active Collaborative Prediction

Dawei Feng[a], Cécile Germain-Renaud[a], Tristan Glatard[b]

[a] *Laboratoire de Recherche en Informatique, CNRS − Université Paris-Sud*
[b] *CREATIS, CNRS − INSERM − Université Lyon 1 − INSA Lyon*

Abstract

Isolating users from the inevitable faults in large distributed systems is critical to Quality of Experience. We formulate the problem of probe selection for fault prediction based on end-to-end probing as a Collaborative Prediction (CP) problem. On an extensive experimental dataset from the EGI grid, the combination of the Maximum Margin Matrix Factorization approach to CP and Active Learning shows excellent performance, reducing the number of probes typically by 80% to 90%. Comparison with other Collaborative Prediction strategies show that Active Probing is most efficient at dealing with the various sources of data variability.

Keywords: Grid; Fault monitoring; Collaborative Prediction; Matrix Factorization; Active Learning

1. Introduction

The recent crash of the Amazon Cloud [1] highlighted the importance of timely discovery of failures in large scale distributed systems: a local, limited error may result in a global catastrophe. Thus a significant part of the software infrastructure of large scale distributed systems, grids or clouds, collects information (monitoring) that will be exploited to discover (knowledge) if, where, and when the system is faulty.

This paper addresses the knowledge building step in the context of end-to-end probing as the class of monitoring techniques. In the end-to-end probing approach, a *probe* is a program launched from a reliable entry point (probe station), which tests the availability (and possibly performance, but this is outside the scope of this paper) of the components on its path. The only observables are the outcomes of the probes, which are binary: success or failure. For instance, a *ping* command would test the overall software stacks and network connectivity from the probe station to the target computer.

The motivating application comes from operations management in the European Grid Initiative (EGI). Challenged by a high fault rate, especially concerning data access, the Biomed Virtual Organization daily runs end-to-end probes to test the availability of all relations between its endpoints, namely Computing Elements (CEs) and Storage Elements (SEs). Our objective is to minimize the number of probes for a given discovery performance target. The probe overhead budget can then be spent on more intelligible probes.

We address the probe minimization problem as an instance of Collaborative Prediction (CP): given a small number of probe results, how to infer the capacities for other (CE, SE) pairs for which no probe is launched. Srebro et al. [2] made a decisive advance in CP by proposing the Maximum Margin Matrix Factorization (MMMF) method. This paper proposes three combinations of probe selection methods with MMMF. Extensive experiments show that the number of probes can be reduced by more than 90%.

Email addresses: Dawei.Feng@lri.fr (Dawei Feng), cecile.germain@lri.fr (Cécile Germain-Renaud), tristan.glatard@creatis.insa-lyon.fr (Tristan Glatard)

This result goes well beyond the particular application: the fundamental hypothesis for CP is that a limited number of common and hidden factors root the outcomes, here success or failure; this hypothesis is reasonable in the context of large scale distributed systems, where the hidden causes are likely to be related to hardware failures and misconfiguration of middleware services shared by clusters of users. Here we extend the results of [3] in order to highlight in which directions the general CP framework should be adapted to this new application area. The key point is that monitoring large scale distributed systems differ from CP's usual applications (personalized recommendation), in two major ways. On the bright side, while users cannot be queried for specific recommendations, probes can be launched at will. On the downside, the distribution of the probe results is highly skewed, faults being a small fraction of the total population. In turn, the unbalanced distributions stem from two origins: firstly, fault causes are hopefully rare; and second, some of the faults are transients. In the recent years, CP methods highlighting the role of various bias have received a lot of attention, partially due to their success in the BellKor solution that won the Netflix challenge [4], and specifically address time variability. In [3], we motivated the choice of MMMF amongst numerous existing CP methods by the fact that it makes the optimization problem both well-defined and tractable. While this is perfectly true, it turns out that the major advantage of MMMF is to be easily amenable to active learning, which addresses fault sparsity both at the spatial (skewed distributions) and temporal (transients) level.

Therefore , the main contributions of this paper are threefold:

- modelling probe-based fault prediction as a CP task;
- experimental evidence that MMMF is an extremely efficient strategy for fault prediction[1];
- comparative analysis motivating the critical advantage of active learning.

The rest of this paper is organized as follows. Section 2 presents the motivating application and discusses the general context of fault prediction and fault diagnosis; section 3 details the probe selection algorithms based on MMMF; the data, evaluation methodology and experimental results for these algorithms are presented is section 4; section 5 discusses related work; section 6 explores two CP methods that can be considered as appealing alternatives to MMMF along the previous discussion; finally, section 7 gives the conclusion.

2. Problem Statement

2.1. Motivating application

The European Grid Infrastructure (EGI) enables access to computing resources for European researchers from all fields of science, including high energy physics, humanities, biology and more. The infrastructure federates some 350 sites world-wide, gathering more than 250,000 cores, which makes it the largest non-profit distributed system in the world. Hardware and software failures are intrinsic to such large-scale systems. Resource availability in production is about 90%, and middleware e.g. gLite [6] , Globus [7] or ARC [8] cannot handle this without substantial human intervention. Access rights to EGI are primarily organized along the concept of Virtual Organization (VO), and each of the 200 VOs has to be specifically configured on its supporting sites, which adds complexity and introduces extra failures. User communities exploit two strategies to cope with faults: overlay middleware e.g. DIRAC [9], DIANE [10], AliEn [11] and PaNDA [12] implements specific fault-tolerance strategies to isolate users from the vagaries of the infrastructure; and monitoring identifies problems and quantifies performance w.r.t. quality of service agreements.

The target system of this work is the Biomed VO. Biomed has access to 256 Computing Elements (CEs) and 121 Storage Elements (SEs). CEs are shares of computing resources, implemented as queues of each site manager (e.g. PBS), and SEs are shares of storage resources; the formal definition is part of the Glue Information model [13]. Testing the availability of all CE-SE pairs is one of the most challenging issues

[1]The datasets are available online at the Grid Observatory [5] portal www.grid-observatory.org, making our experiments reproducible

encountered daily by monitoring operators. The current method is brute force: it periodically launches a fully distributed all-pairs availability test, for a total of 29512 tests, multiplied by the number of capacities to test at each run. Human operators cannot handle so many results; in practice, only a few issues are reported, with questionable selection criteria. With CP, a massive reduction of the number of tests provides nearly similar availability evaluation performance, creating opportunities for better frequency/intrusiveness tradeoff and selection of reported incidents.

2.2. Fault prediction and fault diagnosis

Minimizing the number of probes can be addressed along three avenues: fault *prediction*, *detection* and *diagnosis*. In all cases, the system under consideration is a set of hardware and software components, which can be functioning correctly (UP) or not (DOWN). These components feature some dependencies – e.g. a service certainly depends on the hardware it is running on, and possibly of other services. In the detection problem, the goal is to discover if any of the components is DOWN, while in the diagnosis problem, the goal is to exhibit all DOWN components. Both cases assume a priori knowledge of the components of the system, as well as knowledge of the dependency matrix, which describes the outcome of each probe given the status (UP or DOWN) of these components. For a given set of probes, diagnosis has linear complexity in the number of components; however, optimal probe selection is NP-hard for detection and diagnosis, because it is equivalent to the minimum cover set problem [14].

The obvious advantage of detection and diagnosis is that they provide an explanation of the failure, by exhibiting culprits. On the other hand, it strongly relies on a priori knowledge – which components are required for a probe to succeed – through the dependency matrix. For massively distributed systems, where Lamport's famous definition "A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable" applies, assuming such knowledge might be hazardous in principle. In our case, the very large scale of the dependency matrix, and its particular shape, mainly a block-diagonal structure, calls for further research in order to explore the typical multi-faults configurations.

Instead, this paper focuses on fault prediction. In this case, the overall infrastructure is a black box, with no a priori knowledge of its structure. The question is, given a small number of probe results, how to infer the capacities for other (CE, SE) pairs for which no probe is launched. In this context, fault prediction can be considered as a case for CP. CP is originally a technique for predicting unknown ratings of products for a particular user, based on observed data from other users and products. It can be applied to various domains such as online recommendation, link prediction and distributed system management applications discussed in this paper. It is fundamentally a matrix completion task: given a very sparse user-by-product matrix, whose non-zero entries represent known ratings, predict the unknown entries of the matrix. The success of CP relies on the hypothesis of a *factorial* model: hidden and partially shared factors affect the matrix entries. For example, two nodes (CE or SE) may share several hidden factors - e.g. location, with the associated network connectivity issues, or use of a particular instance of any middleware service (e.g. brokering, authentication), such that the availability of the CE-SE relation may be affected similarly.

Optimal probe selection for fault prediction is NP-hard too: assume that the hidden factors have been identified by CP, and that we want to exhibit the optimal probe set, a posteriori. Then, the probe selection problem is equivalent to diagnosis, but with the factors as components.

3. Goals and methods

From the previous description, minimizing the number of probes encompasses two distinct issues: probe selection, i.e. which subset of the (CE,SE) pairs will actually be tested; and prediction of the availability of all (CE,SE) pairs from the outcome of the previous selection.

3.1. Probe selection

We consider three probe selection methods.

- **Static-Uniform.** The probes are selected uniformly at random amongst all (CE,SE) pairs. In this setting, the probe selection and the prediction are completely independent: the prediction step has no influence over the choice of the probes. This would be unrealistic in recommendation systems (users do not select uniformly the products they rate amongst all proposed), but can be fully implemented in probe selection. Moreover, for the subsequent prediction task, uniform sampling provides theoretical bounds on the MMMF generalization error.

- **Active Probing.** With Active Probing, the set of probes is constructed dynamically, with an initial set of probes selected for instance by the Static-Uniform method, and run through the system to get basic information; then, additional probes are selected and launched with the goal of maximizing some measure of information. Algorithm 1 illustrates the process: a predicted matrix is first given by standard matrix completion based on some pre-selected samples, then some heuristics are used for filtering the next subset of samples, which are labeled by actually running the probes and observing their outcome. After several iterations, a final prediction is returned. In this setting, the CP method impacts the probe selection. In this work, the min-margin heuristic [15] is used for selecting additional probes. Min-margin favors exploration over exploitation: it choses the probe where the uncertainty of the classification result is maximal, and has been demonstrated to be efficient for CP problems [16] .

- **Differentiated costs.** In the two previous methods, the same penalty is associated with both kinds of mispredictions. It might be argued that a false negative (predicting success while the actual result is a failure) is more harmful than a false positive (predicting failure while the actual result is a success), because the federated nature the computational resources offers multiple options to users. Unbalanced costs (in either direction) arise in many other contexts, e.g. medical testing [17], and can be integrated in the core learning step, as shown in the next section.

input : Initial partially observed binary(-1/+1) matrix M_0, threshold λ, max # of new samples N, active-sampling heuristic h
output : Full binary-valued matrix M^{T_i} predicting unobserved entries of M_0
initialize: Initialize the vars
$S(T_0) = S(M_0)$ /*currently observed entries set*/ ;
$i = 0$ /*current iteration times*/ ;
$n = 0$ /*current number of new samples*/ ;
while $(n < N)$ **do**
  $M^{T_i} = StandardMC(S(T_i))$ /*Prediction based on observed entries via standard MC procedure*/ ;
  $S'(T_i) = ActiveSampling(M^{T_i}, h, \lambda)$ /*Actively choose the next set of new samples and query their labels*/ ;
  $S(T_{i+1}) = S(T_i) \cup S'(T_i)$;
  $n = n + \#S'(T_i)$;
  $i = i + 1$;
end

Algorithm 1: Generic active probing algorithm

3.2. Collaborative Prediction with MMMF

This section sketches the motivations and technicalities of MMMF as proposed by Srebro et al. [2]. CP is formalized as a matrix completion problem: if Y is the observed (sparse) matrix, the CP problem is to find a full, real-valued, matrix X of the same size that approximates Y, i.e. that minimizes the "discrepancy" between X and Y without any external information. Assuming a linear factor model, where k hidden factors define the user preference through a linear combination of them, X is constrained to be of rank k. Bounding k to small values (low-rank approximation) does not lead to feasible optimization problems for a partially

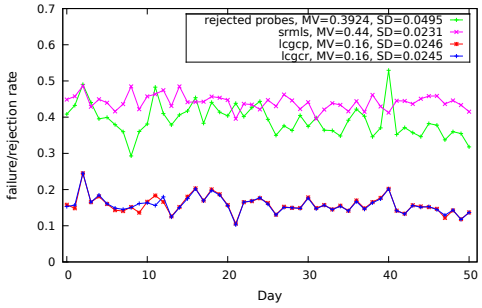


Figure 1: Failure and rejection rates; the mean and standard deviation are computed over the experiments

observed matrix and for the binary setting. The key insight in MMMF is to replace rank minimization with trace norm ($\|X\|_\Sigma$) minimization, under the constraint of no (hard-margin), or small (soft-margin), discrepancy. Let S be the set of known entries in Y. Two objective functions can be considered.

- Hard-margin: minimize $\|X\|_\Sigma$ under the constraints

$$Y_{ij}X_{ij} \geq 1 \text{ for all } ij \in S;$$

- Soft-margin: minimize

$$|X\|_\Sigma + C \sum_{ij \in S} \max(0, 1 - X_{ij}Y_{ij}). \quad (1)$$

As the minimization procedure produces a real-valued matrix, a decision threshold (e.g. positives values give +1, negatives give -1) gives the final predicted binary matrix

The soft-margin factorization can be extended with the general robust strategy described by [18] for integrating differentiated costs (or unbalanced positives and negatives examples) in Support Vector Machines: the regularization parameter C in eq. 1 is split in two, C_+ (resp. C_-) for positive (resp. negative) examples. The only important parameter is the ratio C_+/C_-.

4. Experimental results

4.1. The data sets

Different capabilities have to be tested; in the following, we consider three of them: probe *srm-ls* tests the list ability from a CE to a SE, probe *lcg-cr* tests the read ability from a CE to a SE, and probe *lcg-cp* tests the write ability alike. Thus, each CE works as a probe station, launching probes to test the functionalities between itself and each SE. For the Biomed grid a whole set of testing transactions (as we mentioned before: 29512) were launched each day for each of the three probe classes. After nearly two months running, information for 51 validated days were collected. In other words, 51 fully observed SE-by-CE result matrices were obtained for each probe. Figure 1 shows the statistical profile of the probe outcomes[2]. Failure rates of *lcg-cp* and *lcg-cr* are almost identical (ranges from 10% to 25%), while failure of *srm-ls* is significantly higher (ranges from 40% to 50%).

The probes themselves are gLite jobs, run by a regular Biomed user. Some of them fail (*rejection*) in the sense that gLite is not able to complete the job, denoting that some job management services may be down or misconfigured (e.g. authentication, brokering etc.). The *rejected probes* entries in figure 1 shows the ratio of unsuccessful probes over all launched probes in this sense. In the following, we consider only

[2]Note that here and on Fig. 5 and 6(b), only the points associated to each experiment are meaningful; the lines between the experiments are added only for readability purpose.

the accepted probes, i.e. those which run to completion, reporting success or failure; this approach amounts to consider that the data access capacities are independent from job management. This is a reasonable hypothesis in a gLite infrastructure because file transfers involved in job management use dedicated storage space independent from the one tested by our probes. Separate testing is good practice in general; in this specific case, the high rejection rate (average 40%) and the high failure rate would act as a massive noise on each other, and would make CP more difficult if we tried a global approach.

Table 1: Five example datasets

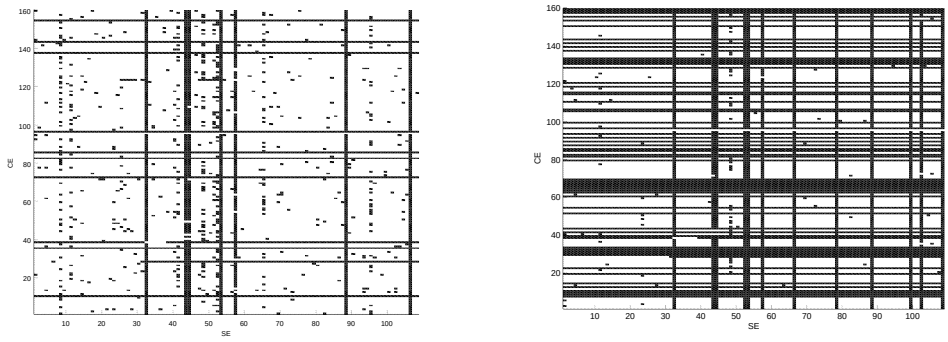
Name	Date	Probe	FAILED Native	FAILED Curated
1	04.21.2011	lcg-cp	0.15	0.04
		lcg-cr	0.16	0.05
		srm-ls	0.45	0.02
2	05.14.2011	lcg-cp	0.14	0.03
		lcg-cr	0.15	0.03
		srm-ls	0.43	0.01
3	05.25.2011	lcg-cp	0.16	0.03
		lcg-cr	0.15	0.03
		srm-ls	0.43	0.02
4	06.09.2011	lcg-cp	0.16	0.05
		lcg-cr	0.16	0.05
		srm-ls	0.42	0.01
5	07.05.2011	lcg-cp	0.16	0.06
		lcg-cr	0.16	0.07
		srm-ls	0.45	0.04

We selected five days for detailed performance analysis, refereed afterwards as the benchmarks. Table 1 shows some basic characteristics of the fully observed entries; the fourth column gives the failure rate when all probes are considered and exemplifies the need for inferring the structure of the apparently massive randomness. Figure 2 illustrates the structure for *lcg-cr* and *srm-ls* on day 5 ('07-05-2011'), where rows represent CEs and columns stand for SEs. Each entry is the probe result between the corresponding CE and SE. Black columns correspond to prolonged SE downtimes while black lines are CE failures leading to complete inability to communicate with any SE (e.g. network downtime or configuration issue). These are usually easily detected and reported by human operators with only a few incident reports. The scattered points correspond to local or transient issues, which are very difficult to handle due to the amount of incident reports independently generated. The higher failure rate of *srm-ls* compared to *lcg-cr* appears to be associated with an inadequate port number in some probes, and may be considered as an example of user error.

It could be argued that other, global (EGI-wide) monitoring tools should report on these systematic failures, and that the probe selection and prediction methods should be applied only to the more elusive causes of errors. While this is disputable (remember that all probes succeed as jobs, thus at least the CEs are up and running), it is worth assessing the performance of the methods when these systematic errors are eliminated. Therefore, we designed a second set of experiments, with *curated* matrices as the reference fault structure. A curated matrix is a new original matrix, where the lines and columns with only failed entries (black ones in figure 2) have been removed prior to analysis. Their basic statistics are shown in the last column of table 1. In this case, *srm-ls* shows a lower error rate than the other probes.

4.2. Evaluation Methodology

From this dataset, evaluating probe selection is straightforward. Figure 3 illustrates the general workflow of the selection-prediction process. The *Original* matrix is the ground truth: a fully observed result matrix



(a) lcg-cp (b) srm-ls

Figure 2: The CE-SE matrix. Black = FAILED, white= OK

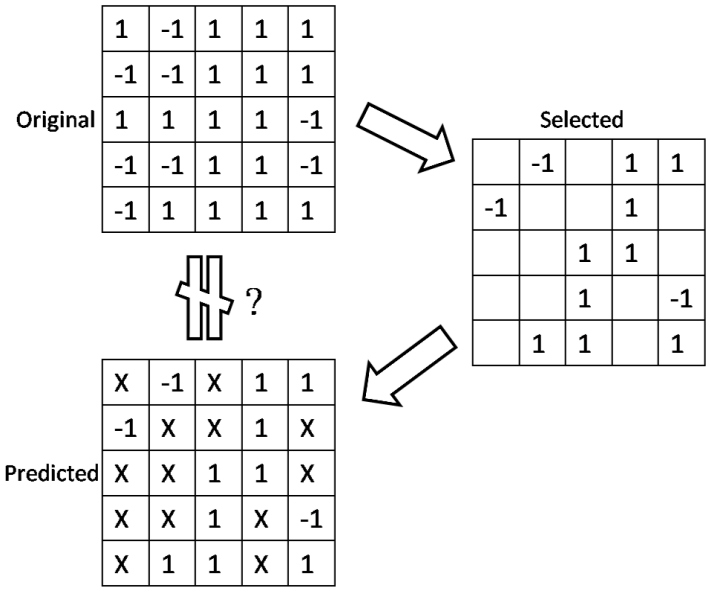
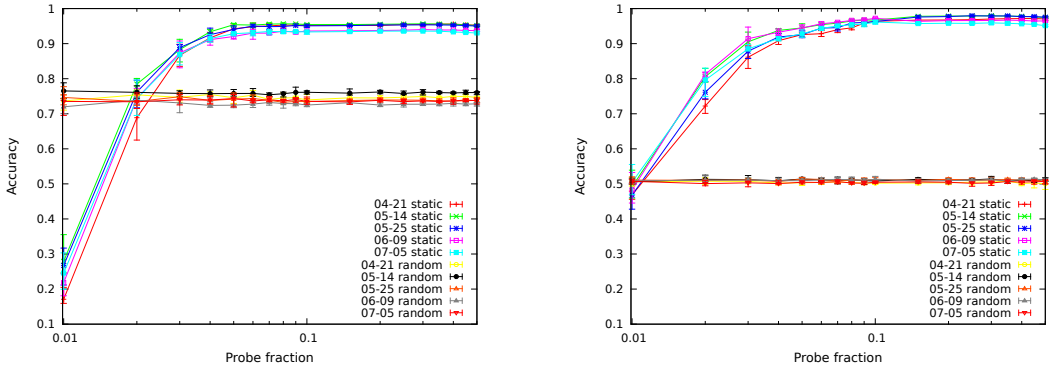


Figure 3: Illustration of matrix recovery



(a) lcg-cp (b) srm-ls

Figure 4: Accuracy for the Static-Uniform probe selection.

obtained from the all-to-all monitoring runs (section 2.1). Value -1 stands for probe result OK (negative) and 1 means FAILED (positive). The *Selected* matrix is generated by deleting a proper proportion of entries in the *Original* one. In a real-world, probe selection-based, monitoring, the remaining entries would be the only actually launched probes. The *Predicted* matrix is the recovery result generated by the prediction algorithm based on the known entries in *Selected*, where the X entries are now set to 1 or -1. In the real-world scenario they would be delivered to users.

Contrary to the recommendation systems, where there is no ground truth (the users do not rate all products), the collection of data presented in section 4.1 provides the true values. Thus, the classical performance indicators for binary classification can be measured. They describe the various facets of the discrepancy between the *Original* and the *Predicted* matrices.

- Accuracy: the ratio of correctly predicted entries over the total number of entries.
- Indicators associated with the risks (confusion matrix): sensitivity, the proportion of actual positives that are correctly predicted; specificity, the proportion of actual negatives that are correctly predicted; precision, the ratio of true positives over all predicted positives, and the MCC (Matthews Correlation Coefficient), a correlation coefficient between the observed and predicted binary classifications that is relatively insensitive to unbalanced positives and negatives.
- The AUC (Area Under ROC Curve), which summarizes the intrinsic quality of a binary classifier independent of the decision threshold.

The interest of MCC and AUC comes from the fact that, in the optimization step of MMMF, the classification error on the *Selected* matrix is a reasonable estimation of the prediction error, while this hypothesis is less natural for estimating MCC and AUC [19]. Thus, MCC and AUC provide a comparison indicator of the performance of the methods beyond their explicit optimization target.

In order to evaluate the contribution of the prediction (or coupled selection-prediction) methods, we compare their results with a simple baseline, called *Rand_Guess* in the following. Rand_Guess predicts entries following the distribution of the sample set (*Selected* matrix). For example if the ratio of positive:negative entries in a sample set is 1:4, then Rand_Guess would predict an unknown entry as *failed* or *positive* with a probability of 20% and as *ok* or *negative* with a probability of 80%.

4.3. Static-Uniform

For each result matrix M different fractions of its entries are deleted uniformly and a series of partially observed matrices M'_1, M'_2, ... are generated. For these new matrices, the task is only to predict the deleted

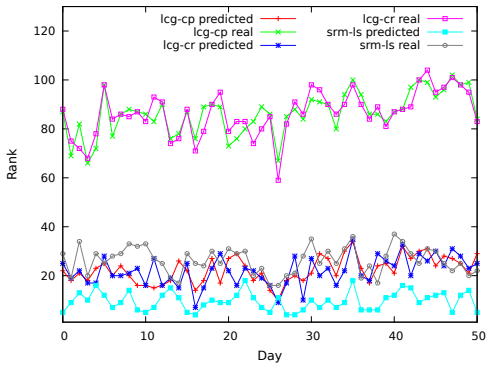
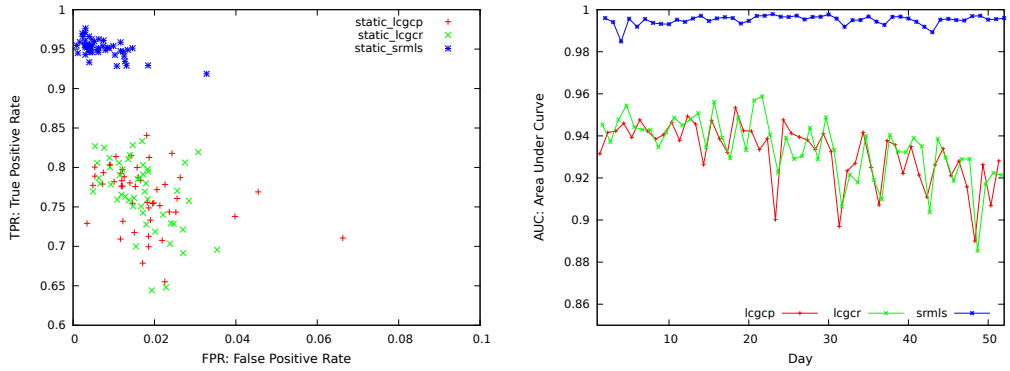


Figure 5: Rank comparison for the Static-Uniform probe selection



(a) ROC space (partial)

(b) AUC

Figure 6: ROC-related metrics, Static-Uniform probe selection

entries from the selected ones by MMMF-based CP. Figure 4 shows the prediction accuracy as a function of the fraction of launched probes, for the five benchmarks. The results are averages over ten experiments. As lcg-cp and lcg-cr behave similarly, only one is shownThe first and striking result is that an excellent performance can be reached with a tiny fraction of the original probes, typically 5%. The *Rand_Guess* results are plotted for comparison purpose, but can be approximated easily: if q is the fraction of positive entries in the original matrix, then in the deleted part, $P(\text{True Positive}) = P(\text{Positive})P(\text{Predicted Positive}) = q^2$, and similarly $P(\text{True Negative}) = (1-q)^2$; overall, the accuracy is $q^2 + (1-q)^2$. With the values of q from table 1, the accuracy of *Rand_Guess* is in the order of 0.7 for *lcg-cp* and *lcg-cr*, and 0.5 for *srm-ls*.

In the CP interpretation, the rank of a result matrix corresponds to the hidden causes. Figure 5 shows the ranks of the predicted and original matrices. The ranks of the predicted matrices are significantly lower than the original ones, showing that a small number of causes dominates the overall behavior. The number of hidden causes is much larger for *lcg-cp* and *lcg-cr* than for *srm-ls*, confirming the empirical evidence that the *srm-ls* faults are more deterministic.

Figure 6(a) is the classical visualization of the confusion matrix in the ROC space for all the 51 days at 90% deletion rate (keeping 10% of the probes). Note the range of the axes, which cover only the small part of the ROC space where the results belong, thus the diagonal line is not visible on the plot. Perfect prediction would yield a point in the upper left corner at coordinate (0,1) of the ROC space, representing 100% sensitivity (no false negatives) and 100% specificity (no false positives). The *srm-ls* dataset shows excellent prediction performance, being mostly very close to (0,1); lcg-cp and lcg-cr exhibit close ROC

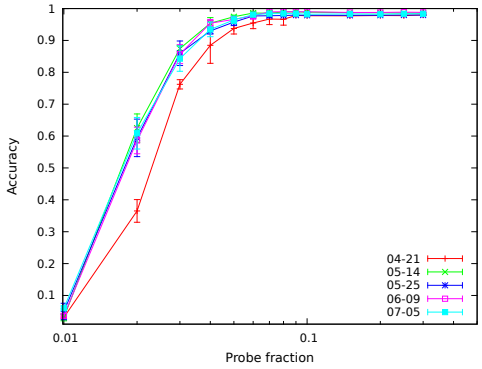


Figure 7: Accuracy for the Static-Uniform probe selection, curated *srm-ls*.

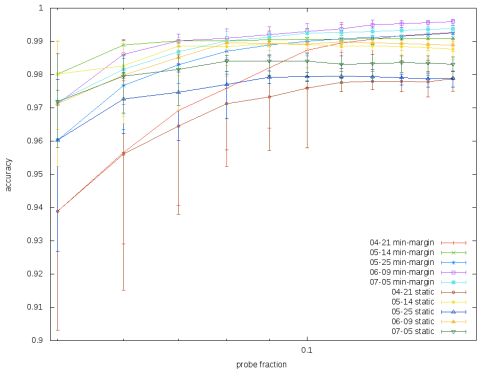


Figure 8: Accuracy comparison between the Static-Uniform and Active probe selection, curated *srm-ls* for the five benchmarks

value distributions, definitely much better than a random guess, which lies on the diagonal line. The other indicators also show excellent performance: the AUC (fig. 6(b)) as well as the MCC are close to 1. The case is closed for the initial problem.

The problem becomes much more difficult when the systematic faults are excluded, thus taking the curated matrices as inputs. Figure 7 shows the prediction accuracy on the curated *srm-ls* example (figures for the other probes are equally excellent, and are omitted; note that, from table 1, this probe is the most challenging one). As before, at most 10% of the whole probes is needed to reach a promising accuracy, greater than 98%. However, as the number of failed entries left in the curated matrices is much less than in the un-curated ones, e.g. the fraction of failed entries on day 1 (*srm-ls, 04-21-2011*) drops from 45.37% to 2.25%, accuracy is not meaningful: predicting all entries as negative would give a similar result. The ability of making good prediction on the *failed* entries should be valued more. And the relevant performance indicators are not so good, except for day 5, as shown in figure 8: for the same example, at 10% deletion rate, sensitivity is 0.32, meaning that 68% of the failures are not predicted, and precision is 0.49, meaning that amongst the predicted failures, 51% are spurious. The first strategy to tackle this issue is Active Probing.

4.4. Active Probing

In this experiment, we compare the Active Probing strategy with the Static one at equal probing cost: first, a Static-Uniform method is applied, in order to get the reference information, then more probes are

selected with the min-margin heuristic for Active Probing, while for the Static-Uniform method, the same number of probes are selected uniformly at random.

Active Probing does improve accuracy over Static-Uniform, as shown in figure 8. However, as explained in the previous section, the quality of failure prediction is the most important goal in this context. Figure 9 compares the relevant indicators: sensitivity, precision and the MCC. They are detailed for the initial probe fraction equal to 5%, then adding probes by step of 5% fractions. The results as given for a total of 10% and 15% probes. The first result is that Active Probing always outperforms Static-Uniform. More importantly, acceptable results can be obtained with a relatively small number of probes (15%), albeit larger than in the much easier un-curated case: in all cases, more than 90% of predicted failures are actual ones (figure 9(b)), even for the very difficult day 2; the probability of predicting an actual failure (figure 9(a)) increases from 43% to 67% on day 1, from 39% to 62% on day 3 and from 14% to 48% on day 4. In other words, and as expected, Active Probing singled out the failures as the most uncertain data, adaptively building its own training set.

The performance greatly varies with the benchmark, and the variation is somehow related to the failure rate of the benchmark (table 1): larger failure rates in the original curated matrix help uncovering the structure of the faults, even at quite low levels: with 4% failure rate, the *07-05-2011* (day 5) benchmark exhibits acceptable performance when keeping only 5% of the probes and the Static-Uniform strategy; conversely, for day 2, with a low failure rate (1%), sensitivity remains bad, predicting at best 19% of the actual faults, although active probing allows for a good precision. However, the failure rate does not tell the full story: days 2 and 4 have the same low one, but the performance on day 4 is much better. The likely explanation is that faults on day 2 do not present much correlation, while faults on day 4 derive from a small number of shared causes.

4.5. Cost sensitive + Active probing

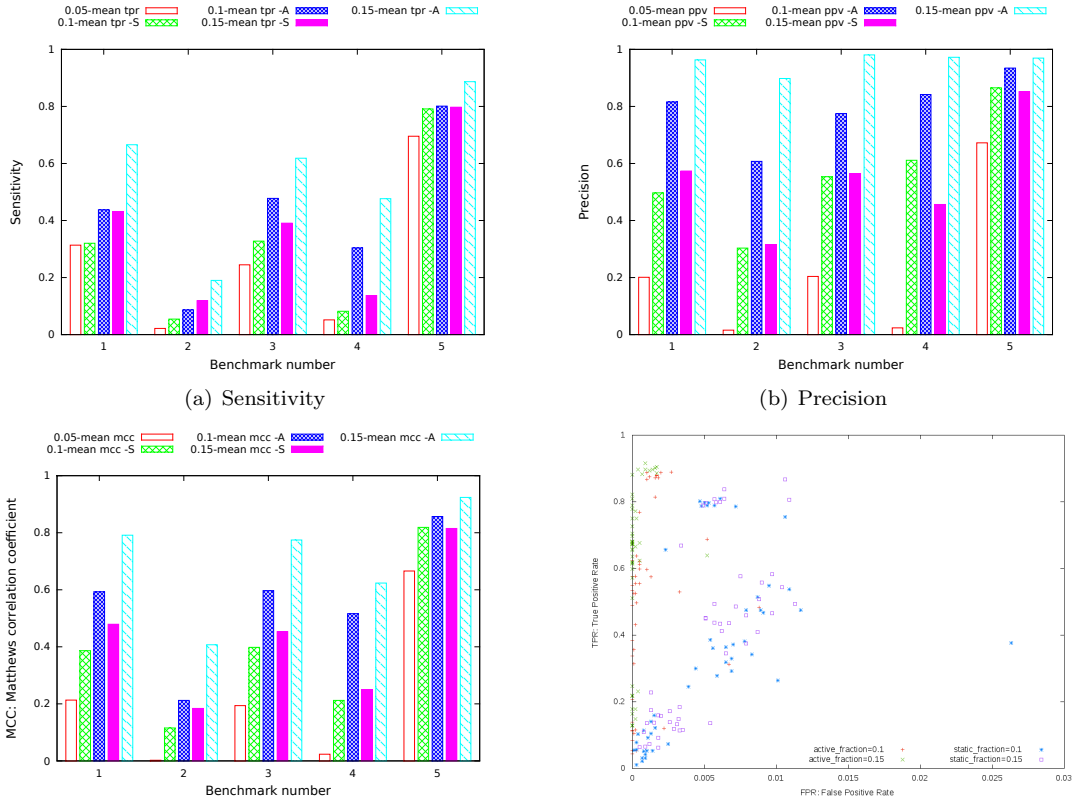
Finally, we sketch the results of the cost-sensitive MMMF. The C^+/C^- ratio is set equal to 10. The optimization target being soft-margin, the results for the initial Static-Uniform at 5% probe fraction are slightly different from the previous experiments. Figure 10 compares Active probing with and without cost weighting. Higher penalization of false negatives almost always decreases the final mis-prediction costs (Figure 10 (a)). Figure 10 (b) and (c) give the explanation: while sensitivity is indeed increased, the number of false positives also increases, leading to slightly lower precision, but the overall impact is favorable.

4.6. Computational cost

CP methods have to be scalable, as they target enormous data sets such as the Netflix database. The computational cost of the optimization problem of learning a MMMF essentially depends on the number of known entries in the *Selected* matrix, or equivalently on the probe fraction. Technically, the optimization is performed through a sparse dual semi-definite program (SDP), with the number of variables equal to the number of observed entries. We used YALMIP [20] as the model tool and CSDP [21] as the SDP solver. Empirically, the time needed for computing one MMMF increases exponentially with the number of entries in the *Selected* matrix. In practice, computation time is not an issue: less than 30 seconds with 2000 entries (15% probes) on a standard workstation.

5. Related Work

Collaborative Prediction associated with end-to-end probing, with the components structure considered as a black box, participates in the general Quality of Experience approach [22]. More precisely, an important ingredient separating QoE from QoS is binary (possibly extended to discrete) classification. Most work in this area is devoted to network-based services (e.g. among many others [23]). Before QoE became a popular keyword, Rish and Tesauro [16] explored the combination of MMMF and Active probing for the selection of good servers in various distributed and P2P systems. Our work combines the goal of proposing fault-free services to the user exemplified in [14, 24], and the CP approach of [16]. Z. Zheng and M.R. Lyu propose explicit users collaboration for estimating failure probabilities [24]; while their end-to-end framework is



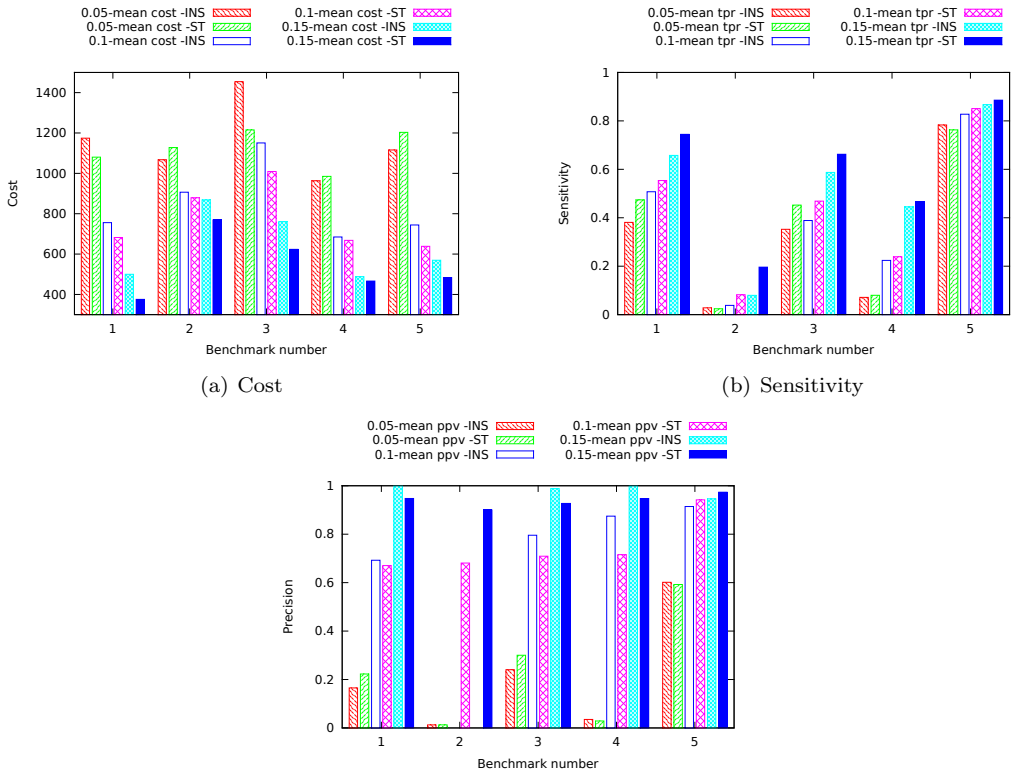
(a) Sensitivity

(b) Precision

(c) MCC

(d) ROC space

Figure 9: Performance comparison for Static-uniform and Active Probing, curated *srm-ls*; , (a), (b) and (c) for the five benchmarks, (d) for all days.



(a) Cost

(b) Sensitivity

(c) Precision

Figure 10: Cost and performance comparison between the Cost-Sensitive and Cost-Insensitive active probing, curated *srm-ls* for the five benchmarks.

related to ours, the goal is more in line with QoS (estimating the full distribution of probability instead of binary classification), and the correlation amongst users or services is modeled by ad hoc methods. To our knowledge, this paper is the first one to exemplify MMMF on fault prediction.

The state of the art methods in Collaborative Prediction follow two paths. Matrix Factorization methods [2, 25, 26], exemplified by MMMF, considers that all data are equally relevant to the prediction task: the values are generated using the same factor vector. This approach, which assumes a strong homogeneity of the data, has been termed *continuous latent factor* by [27]. The other approach considers that the data should be contextualized, and *bias* terms included in the model. The most frequent justification for such contextualization is temporal variation; in recommendation systems, the context can be the mood of the user, or even the fact that two users share a common internet access.

Y. Koren and the BellKor team, winners of the highly published Netflix prize, proposed explicit modeling of various contexts, including temporal drifts and spikes [28]. They remarked that a basic model, which captures context effects but disregards user-item interactions, explains more of the data variability than the commercial Netflix Cinematch recommender system. However, the model selection for bias is largely based on the specificities of users behavior in movie recommendation, thus not easily extendable. Alternatively, *block models* [29, 30] propose a fully generative model and leverage the older neighborhood method [31] based onclustering. The specific application to CP is Bi-LDA [29]. Although LDA (Latent Dirichlet Allocation) also embodies the discovery of latent (hidden) factors, the key difference between Bi-LDA and matrix factorization is that the relationship (CE/SE in our case, user/product in recommendation systems) is allowed to select a new topic (factor) for each interaction.

In our grid environment, transient failures are well attested, and the frequency of switching between functioning/malfunctioning can be high; such context bias could be anything like a middleware installed on a CE temporary down, or too many concurrent write requests issued to a SE, making context-awareness appealing. However, Bi-LDA is known to exhibit relatively poor predictive performance, probably due to an exclusive modeling of interaction of clusters (through topics); in other words, the expression of the specificity of individual interactions (this particular user/CE with this particular movie/SE) is lost. Recently, Mackey proposed a bayesian approach to reconcile Matrix Factorization and probabilistic topic selection with Mixed Membership Factorization (M^3F) [27], introducing context dependence in a more general way than the a priori formulation of [32]. Moreover, in recommendation systems, the greatest performance improvements with M^3F occur for the high-variance, sparsely-rated objects, suggesting a good capacity at capturing the transients that are a serious issue for VO operation managers.

6. The role of Active Learning

6.1. AUC optimization within MMMF

In section 4.4, we have exposed the intuitive motivation for Active Probing: we think that one important issue of CP applied to fault prediction lies in the strong imbalance between positive and negative examples; on the other hand, MMMF is theoretically grounded only for uniform random selection of examples, which is just the opposite of the active learning approach. Thus, a first question is to which extent a more targeted algorithm would not successfully compete with Active Probing. In order to evaluate its specific contribution, we designed an algorithm which integrates MMMF and optimization of the area under the ROC curve (AUC), a natural and useful performance measure for evaluating classifiers when the class distributions are heavily skewed.

Reformulation of the objective function

Intuitively, AUC expresses the probability that a decision function f assigns a higher value to a randomly selected positive example x^+ than to a randomly selected negative example x^-:

$$\mathrm{AUC}(f) = Pr(f(x^+) > f(x^-)).$$

AUC refers to the true distribution of positive and negative instances, and can be estimated through sampling. The normalized Wilcoxon-Mam-Whitney statistic gives the maximum likelihood estimate of the true

AUC given n^+ positive and n^- negative examples [33] :

$$\widehat{AUC}(f) = \frac{\sum_{i=1}^{n^+}\sum_{j=1}^{n^-} 1_{f(x_i^+)>f(x_j^-)}}{n^+n^-}. \tag{2}$$

The AUC score is determined by the number of correctly ranked sample pairs; therefore, to maximize the AUC we could maximize the number of correctly ranked sample pairs, which meet $f(x^+) > f(x^-)$). Here we extend the standard Maximum Margin Matrix Factorization (MMMF) with the object of AUC optimization.

In the MMMF problem, we use the partially observed sparse matrix Y to recover the target matrix X under the constrain of a L_2 norm discrepancy for each predicted and observed entry, i.e $Y_{ij}X_{ij} \geq 1 - \xi$. However, this entry-wise constraint contains no order information between sample pairs, i.e. order between the pair $\langle Y_i, Y_j\rangle$ where $i \neq j$ and $Y_i \in S^+, Y_j \in S^-$. Here, in the aim of AUC score maximization, we add the sample pairwise order constraints to the MMMF objective function and derive the following reformulation.

$$\begin{aligned} min\ \ & \|X\|_\Sigma + \lambda_1 \sum_{k\in S} \xi_k + \lambda_2 \sum_{i\in S^+, j\in S^-} \delta_{ij} \\ s.t.\ \ & Y_k X_k \geq 1 - \xi_k \\ & Y_i^+ X_i^+ + Y_j^- X_j^- \geq 1 - \delta_{ij} \end{aligned}$$

where S is the set of known entries in Y, S^+ and S^- are the positive and negative entry sets, ξ_k is the entry-wise constraint on X_k, δ_{ij} is the pairwise order constraint on $\langle X_i, X_j\rangle$, λ_1 and λ_2 are the regularization terms. One thing to mention is that the number of constraints in the second regularization term is quadratic with the sample size, thus leading to a more complex optimization problem. Inspired by the idea in [34], instead of adding all constraints at once, we add the most important constraint iteratively, with the price of iterative computation. However, in practice, the total number of added constraints on all test sets proved to be quite limited, never exceeding 12, in accordance with a similar observation in [34].

Algorithm Framework

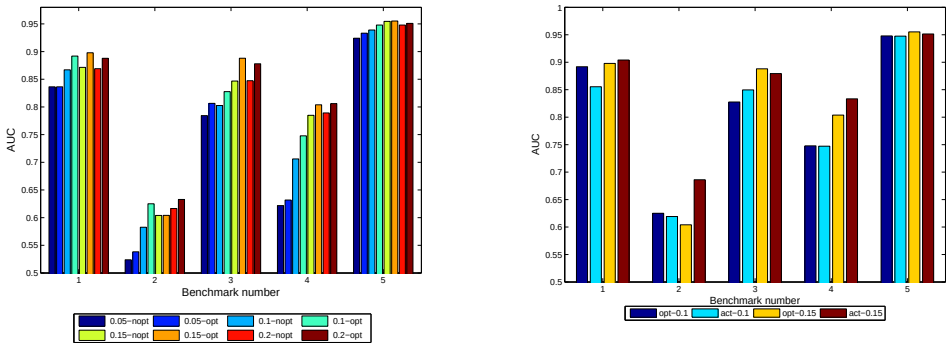
input : Initial partially observed binary(-1/+1) matrix M_0, max number of iteration N
output : Full binary-valued matrix M^{T_i} predicting unobserved entries of M_0
initialize: Initialize the vars
$S(T_0) = S(M_0)$ /*current constraint set*/ ;
$i = 0$ /*current iteration times*/ ;
while $(i < N)$ **do**
  $M^{T_i} = StandardMC(S(T_i))$ /*Prediction based on observed entries via standard MC procedure*/ ;
  $S'(T_i) = MostViolatedAUC(M^{T_i})$ /*Calculate and select the most violated AUC pair */ ;
  if $\#S'(T_i) > 0$ **then**
    $S(T_{i+1}) = S(T_i) \cup S'(T_i)$;
    $i = i + 1$;
  else
    break /*No violated AUC pair*/ ;
  end
end

Algorithm 2: AUC optimization within Matrix Factorization

Algorithm 2 illustrates the process of AUC-oriented MMMF (AUC-MMMF). We use a standard MMMF procedure for the recovery of a partially observed matrix, then the AUC value on the training set is computed according to equation 2 and the most violated AUC pair is added into the current constraint set for the

next iteration. The loop terminates when there is no more violated pairs in the sample set or the maximum number of iteration is reached.

Experimental Results



(a) Static vs AUC optimization

(b) Active vs AUC optimization

Figure 11: AUC optimization on *srm-ls*, with $\lambda_1 = 0.5, \lambda_2 = 10$.

The trade-off coefficients λ_1 and λ_2 are chosen via cross-validation. We ran each experiment 5 times and average the results correspondingly. The performance of AUC-MMMF with static and active MMMF are compared at different sample rate levels on the curated *srm-ls* benchmarks. Figure 11(a) shows that AUC-MMMF always outperforms static MMMF, by about $\%3 - \%6$ on benchmarks $1, 2, 3, 4$. However, Active Probing outperforms or is equivalent to AUC-MMMF in most cases. In other words, the Active Probing strategy actually discovers the violated constraints through focusing on the most uncertain -and very often positive (failure)- cases.

6.2. Mixed Membership Matrix Factorization

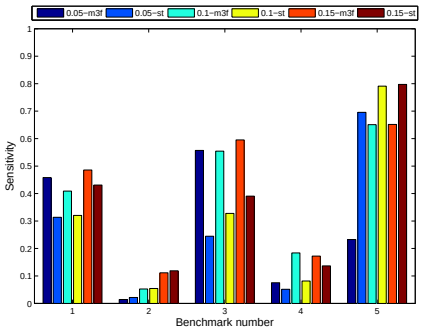
This section compares the performance of one implementation of the Mixed Membership Matrix Factorization with static and active MMMF on the curated probing dataset. For better understandability, we describe the M^3F Topic-Index Bias (TIB) model of [27] in recommendation terms. The model is an implementation of M^3F where the context bias can be additively decomposed into a user bias and a item bias. Both bias are influenced by counterpart's selected topic, i.e. the user bias is influenced by the item's topic and vice versa. In M^{3}F-TIB each user and each item has its own latent factor vectors (a_u and b_j) and topic distribution parameters (θ_u^U and θ_j^M). To rate an item, first both the involved user and item draw a topic, z_{uj}^U for user side topic and z_{uj}^M for item side topic, from their distributions. Then, a rating bias, β_{uj}^{ik}, is jointly specified by the user and item topics, i and k, and the identity of the user and item, u and j. Last, a complete rating is given by the sum of a user-item-specific static rating $a_u \cdot b_j$ and a contextual bias β_{uj}^{ik}, along with some noise. For simplicity, a rating r can be expressed as following:

$$r_{uj} \sim N(\beta_{uj}^{ik} + a_u \cdot b_j, \sigma^2)$$

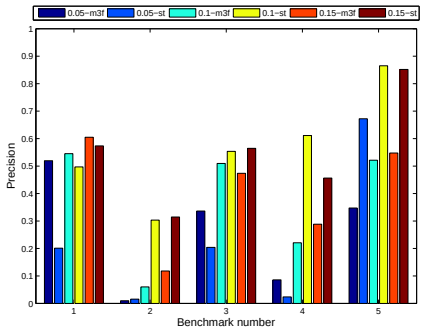
$$\beta_{uj}^{ik} = \chi_0 + c_u^k + d_j^i$$

where σ is a Gaussian noise and χ_0 is a fixed global bias, c_u^k is the bias for user u under item topic j and d_j^i is the bias for item j under user topic i.

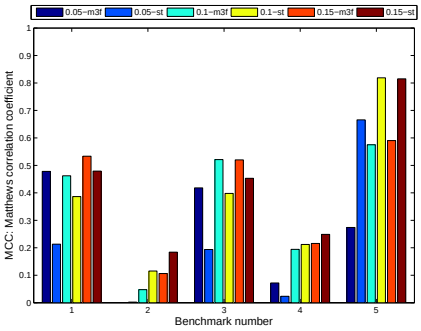
For M^{3}F-TIB, we used a dynamic threshold for labeling the predicted real-valued matrix, as predicted values provided by M^{3}F-TIB are nearly always negative, which makes a fixed threshold like 0 unreasonable. More precisely, we choose the threshold which assigns the final label of each predicted entry as following:



(a) Sensitivity

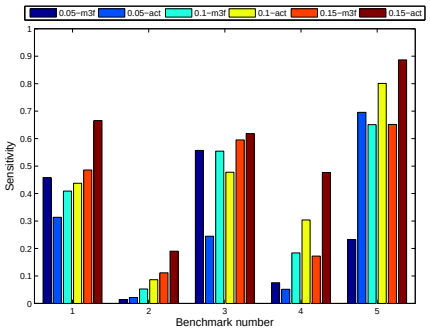


(b) Precision

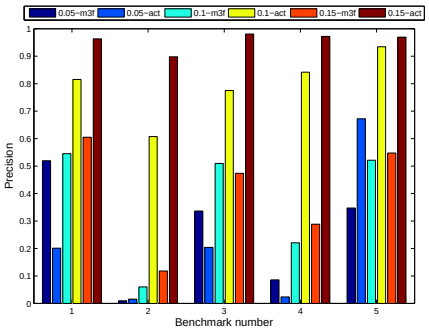


(c) MCC

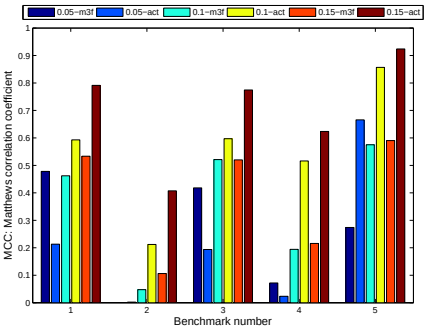
Figure 12: M^{3}F-TIB vs. static MMMF, m3f stands for M^{3}F-TIB and st means static MMMF.



(a) Sensitivity



(b) Precision



(c) MCC

Figure 13: M^{3}F-TIB vs. active MMMF, m3f stands for M^{3}F-TIB and act means active MMMF.

first the proportion of positive samples, $\theta = S^+/S$, is calculated from the training set, then for all the predicted values, we choose the first θ percent as positive entries and the other part as negative ones, in other words assuming that the fraction of positive entries in the training set should approximate the one in the whole set. For M^{3}F-TIB, the model parameters are set as following: $numFacs = 20$, $KU = 2$, $KM = 2$, the Gibbs sampler is initialized using a Maximum A Posteriori (MAP) estimator, and run 500 samples for prediction, without any discarded samples for 'burn-in'. All results are averaged over 10 runs.

Figure 12 illustrates the comparison of different classifier measurements between M^{3}F-TIB and static MMMF. On benchmark 1 and 3, M^{3}F-TIB shows a better performance than the static MMMF over all three metrics, except that the precision of M^{3}F-TIB on benchmark 3 is slightly lower than with static MMMF when the size of training set increases. At the same time sensitivity is significantly better than with MMMF, which implies that the dynamic threshold based M^{3}F-TIB tends to have less false negative than false positives. On the other hand, static MMMF performs better than M^{3}F-TIB on benchmark $2, 4, 5$. Interestingly, on benchmark 4, both algorithm tend to have similar MCC values, but behave oppositely on sensitivity and precision: M^{3}F-TIB has better sensitivity while static MMMF has better precision.

Similar comparison of these metrics between M^{3}F-TIB and active MMMF is demonstrated in figure 13, showing that M^{3}F-TIB is less competitive than active MMMF on all benchmarks. This was to be expected, as M^{3}F-TIB is essentially comparable to static MMMF in our case.

In conclusion, given the operational goal as they are - essentially low intrusiveness probing - the capacity of actively selecting the most informative probes provides a much more efficient method to capture the time variability than M^{3}F.

7. Conclusion

The Achille's heel of large scale production grids and clouds is reliability. At the scale of these systems, classical detection or diagnosis would require a complete a priori knowledge of the software and hardware infrastructures that might remain definitely inaccessible. Quality of Experience at reasonable human cost requires extracting the hidden information from monitoring data whose intrusiveness should be limited. Collaborative Prediction is one of the promising and scalable strategies that can address this new challenge. Compared with the recommendation context, monitoring enjoys a decisive advantage, being allowed to adaptively build knowledge. Through experiments on a large dataset, this paper demonstrates the effectiveness of combining Collaborative Prediction and Active Leaning. The min margin heuristic has shown to be versatile enough to address two difficult issues quite specific to the fault prediction problem, and of very different nature: the spatial one, the imbalance of positive and negative examples; and the temporal one, the transients. Cleaning the data - eliminating the noise -, for instance through bias modeling, has been shown essential for recommendation systems. In our active monitoring setting, things go in the reverse way, to the same goal: instead of acquiring all data, then discarding the most noisy of them, which would be for example the result of a fixed frequency probing strategy, Active Probing adaptively adjusts its acquisitions and its the internal model. The next step would be to go for a fully personalized recommendation system, taking into account not only the infrastructure, but the particular user, whose specificities may also create failure risks.

Acknowledgments

This work has been partially supported by the European Infrastructure Project EGI-InsPIRE INFSO-RI-261323, by France-Grilles, and by the Chinese Scholarship Council.

References

[1] H. Blodget, Amazon's cloud crash disaster permanently destroyed many customers' data (April 2011). URL `http://www.businessinsider.com/amazon-lost-data-2011-4`

[2] N. Srebro, J. D. M. Rennie, T. S. Jaakola, Maximum-margin matrix factorization, in: Advances in Neural Information Processing Systems 17, 2005, pp. 1329–1336.

[3] D. Feng, C. Germain-Renaud, T. Glatard, Distributed monitoring with collaborative prediction, in: 12th Int. Symp. On Cluster, Cloud and Grid Computing, 2012, pp. 376–383.
[4] Y. Koren, The BellKor Solution to the Netflix Grand Prize, Tech. rep., Yahoo! Research (2009).
[5] C. Germain-Renaud, A. Cady, P. Gauron, M. Jouvin, C. Loomis, J. Martyniak, J. Nauroy, G. Philippon, M. Sebag, The Grid Observatory, in: 11th IEEE/ACM Int. Symp. on Cluster, Cloud, and Grid Computing, 2011, pp. 114–123.
[6] E. Laure, al, Programming the Grid with gLite (2006).
[7] I. Foster, The globus toolkit for grid computing, in: IEEE Int. Symp. on Cluster Computing and the Grid, 2001.
[8] M. Ellert, al., Advanced resource connector middleware for lightweight computational grids, Future Generation Computer Systems 23 (2) (2007) 219 – 240.
[9] A. Tsaregorodtsev, al., DIRAC3 . The New Generation of the LHCb Grid Software, Journal of Physics: Conference Series 219 (6) (2009) 062029.
[10] J. T. Moscicki, Diane - distributed analysis environment for grid-enabled simulation and analysis of physics data, in: Nuclear Science Symposium Conference Record, 2003 IEEE, Vol. 3, 2003, pp. 1617–1620 Vol.3.
[11] S. Bagnasco, al., Alien: Alice environment on the grid, Journal of Physics: Conference Series 119 (6) (2008) 062012.
[12] T. Maeno, Panda: distributed production and distributed analysis system for atlas, Journal of Physics: Conference Series 119 (6) (2008) 062036.
[13] S. Andreozzi, al., Glue Schema Specification, V.2.0, Tech. rep. (2009).
[14] I. Rish, al., Adaptive diagnosis in distributed systems, IEEE Trans. Neural Networks 16 (5) (2005) 1088 – 1109.
[15] S. Tong, D. Koller, Support vector machine active learning with applications to text classification, J. Mach. Learn. Res. 2.
[16] I. Rish, G. Tesauro, Estimating end-to-end performance by collaborative prediction with active sampling, in: Integrated Network Management, 2007, pp. 294–303.
[17] K. Morik, P. Brockhausen, T. Joachims, Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring, in: 16th Int. Conf. on Machine Learning, 1999, pp. 268–277.
[18] Y. Lin, G. Wahba, H. Zhang, Y. Lee, Statistical properties and adaptive tuning of support vector machines, Machine Learning 48 (2002) 115–136.
[19] T. Joachims, A support vector method for multivariate performance measures, in: International Conference on Machine Learning (ICML), 2005, pp. 377–384.
[20] J. Lfberg, Yalmip : A toolbox for modeling and optimization in MATLAB.
URL `http://users.isy.liu.se/johanl/yalmip`
[21] B. Borchers., Csdp, a c library for semidefinite programming., Optimization Methods and Software 11(1) (1999) 613–623.
[22] H. Rifai, S. Mohammed, A. Mellouk, A brief synthesis of QoS-QoE methodologies, in: 10th Int. Symp. on Programming and Systems, 2011, pp. 32–38.
[23] H. Tran, A. Mellouk, Qoe model driven for network services, in: Wired/Wireless Internet Communications, Vol. 6074 of LNCS, 2010, pp. 264–277.
[24] Z. Zheng, M. R. Lyu, Collaborative reliability prediction of service-oriented systems, in: 32nd ACM/IEEE Int. Conf. on Software Engineering, 2010.
[25] R. Salakhutdinov, N. Srebro, Collaborative filtering in a non-uniform world: Learning with the weighted trace norm, in: 24th Conference on Neural Information Processing Systems (NIPS), 2010, pp. 2056–2064.
[26] G. Takács, I. Pilászy, B. Németh, D. Tikk, Scalable collaborative filtering approaches for large recommender systems, Journal of Machine Learning Research (JMLR) 10 (2009) 623–656.
[27] L. W. Mackey, D. Weiss, M. I. Jordan, Mixed membership matrix factorization, in: 27th International Conference on Machine Learning (ICML-10), 2010.
[28] Y. Koren, Collaborative Filtering with Temporal Dynamics, in: 15th ACM SIGKDD international conference on Knowledge Discovery and Data Mining, 2009, pp. 447–456.
[29] I. Porteous, E. Bart, M. Welling, Multi-hdp: a non parametric bayesian model for tensor factorization, in: 23rd Conf. on Artificial Intelligence, 2008, pp. 1487–1490.
[30] E. M. Airoldi, D. M. Blei, S. E. Fienberg, E. P. Xing, Mixed membership stochastic blockmodels, Journal of Machine Learning Research (JMLR) 9 (2008) 1981–2014.
[31] J. L. Herlocker, J. A. Konstan, A. Borchers, J. Riedl, An algorithmic framework for performing collaborative filtering, in: 22nd ACM SIGIR conference on Research and development in information retrieval, 1999, pp. 230–237.
[32] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in: 14th ACM SIGKDD int. conf. on Knowledge discovery and data mining, 2008, pp. 426–434.
[33] L. Yan, R. H. Dodier, M. Mozer, R. H. Wolniewicz, Optimizing Classifier Performance via an Approximation to the Wilcoxon-Mann-Whitney Statistic, in: 20th International Conference on Machine Learning (ICML-03), 2003, pp. 848–855.
[34] I. Tsochantaridis, T. Joachims, T. Hofmann, Y. Altun, Large margin methods for structured and interdependent output variables, Journal of Machine Learning Research (JMLR) 6 (2005) 1453–1484.