Response Prediction Using Collaborative Filtering with Hierarchies and Side-information

Aditya Krishna Menon*
University of California, San Diego
La Jolla, CA, USA
akmenon@ucsd.edu

Krishna-Prasad Chitrapura
Yahoo! Labs
Bengaluru, Karnataka, India
pkrishna@yahoo-inc.com

Sachin Garg
Yahoo! Labs
Bengaluru, Karnataka, India
gsachin@yahoo-inc.com

Deepak Agarwal
Yahoo! Research
Santa Clara, CA, USA
dagarwal@yahoo-inc.com

Nagaraj Kota
Yahoo! Labs
Bengaluru, Karnataka, India
nagarajk@yahoo-inc.com

ABSTRACT

In online advertising, response prediction is the problem of estimating the probability that an advertisement is clicked when displayed on a content publisher's webpage. In this paper, we show how response prediction can be viewed as a problem of *matrix completion*, and propose to solve it using matrix factorization techniques from *collaborative filtering* (CF). We point out the two crucial differences between standard CF problems and response prediction, namely the requirement of predicting probabilities rather than scores, and the issue of confidence in matrix entries. We address these issues using a matrix factorization analogue of logistic regression, and by applying a principled confidence-weighting scheme to its objective. We show how this factorization can be seamlessly combined with explicit features or *side-information* for pages and ads, which let us combine the benefits of both approaches. Finally, we combat the extreme sparsity of response prediction data by incorporating *hierarchical information* about the pages and ads into our factorization model. Experiments on three very large real-world datasets show that our model outperforms current state-of-the-art methods for response prediction.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: Online Information Services

General Terms

Algorithms, Design, Experimentation

Keywords

Response prediction, collaborative filtering, hierarchies

*Work done while first author was interning at Yahoo! Bangalore.

KDD'11, August 21–24, 2011, San Diego, California, USA.

1. THE RESPONSE PREDICTION PROBLEM

Online advertising involves the interaction between two entities: *publishers* (e.g. AOL, Yahoo), who own and manage webpages, and *advertisers* (e.g. Pepsi, Nike), who have products that they wish to market to users of the publishers' webpages via ads. A fundamental monetary paradigm in this form of advertising is *pay-per-click*. Here, each advertiser places a bid for their ad to be placed on a publisher's webpage, and pays the bid amount to the publisher only if the ad was selected to be displayed (referred to as a *view*) and subsequently clicked by a user. Amongst many competing ads, the publisher chooses to display the ad with the highest *expected revenue*, which is the ad's bid amount multiplied by the probability that it is clicked. This probability is known as the clickthrough rate (CTR) of an ad, and its reliable estimation is crucial for a publisher to maximize revenue. The task of estimating this probability is known as *response prediction*, or *CTR estimation*.

Response prediction is a challenging problem for at least two reasons. First, the majority of ads have limited or no past history on a particular publisher page. This *sparsity* obviously hinders simple CTR estimation, and necessitates a principled way to exploit correlations in the data. Second, as most ads are clicked very infrequently, we have to predict the probability of a *rare event*, which is very challenging. In both situations, it is beneficial to exploit known *hierarchical information* about pages and ads. This refers to a pre-defined hierarchy for both entities, where for example, the set of all ads is partitioned according to the campaigns they belong to, the campaigns are in turn divided according to the advertisers running the campaigns and so on. (See Figure 1 for an example.) These hierarchies encode correlations between the CTRs – for example, two ads shown as part of the same campaign will have similar CTRs – and so are very useful in deriving reliable probability estimates when historical data is limited.

Existing methods for response prediction are based either on training standard classifiers on explicit features for pages and ads, or on statistical smoothening over baseline estimates. In this paper, we propose a novel approach to the problem based on *collaborative filtering* (CF). This is a technique used in recommender systems, where the input is a matrix of user-by-item preference scores, where most entries are missing. Each non-missing entry is a numeric score telling us how much a user likes an item. The desired output is a set of predicted scores for the missing entries, so that we can recommend for each user some new items that she might

like. We show the natural connection between this problem and response prediction, where we think of pages as users, ads as items, and the entries of the matrix as the historical CTRs for (page, ad) pairs, while pointing out two differences between the problems: (i) response prediction requires the predicted CTRs to be meaningful probabilities, and (ii) our goal in response prediction is to not only fill-in the missing CTRs in this matrix, but also *smoothen* the CTRs for which there is limited historical data. This requires respecting the *confidence* in the matrix entries. Further, from a modelling perspective, the aforementioned problem of data sparsity in response prediction necessitates a careful mechanism for exploiting all available information. To address these differences, we begin by showing how a *matrix factorization* model can meet the basic requirements for response prediction, using simple extensions that have been studied in the CF literature. This extended factorization model is capable of incorporating explicit page and ad features, known in the CF literature as *side-information*. We give a simple iterative scheme by which the side-information can be used to refine our model's performance. We then show how the factorization model can be made to exploit *hierarchical information* about pages and ads. This novel extension allows the model to mitigate the challenge of data sparsity, and constitutes the main contribution of the paper. Our resulting model may be optimized using stochastic gradient descent, allowing it to scale to datasets with several billion matrix entries. Experiments on three real-world datasets show that our model improves on the performance of state-of-the-art response prediction models, and demonstrates the model's scalability.

2. BACKGROUND AND RELATED WORK

There are three main relevant areas of research for this paper: response prediction, hierarchical constraints and collaborative filtering. Before proceeding, we fix the notation used in this paper.

2.1 Notation and terminology

Given a matrix $X \in \mathbb{R}^{m \times n}$, we denote the (i,j)th entry of the matrix by X_{ij} and the jth column of the matrix by X_j. We call each pair (i,j) a *dyad*. We denote the Frobenius norm of a matrix by $||X||_F := \sqrt{\sum_i ||X_i||_2^2}$. If X has entries that are unobserved or missing, we denote these by "?". We let $\mathcal{O} = \{(i,j) : X_{ij} \neq \text{"?"}\}$ be the set of observed entries for the matrix X.

Response prediction. We are interested in predicting probabilities given the interaction between (web)pages $\mathcal{P} = \{1, \ldots, m\}$ and ads $\mathcal{A} = \{1, \ldots, n\}$, where each entity is represented by a unique identifier. Given a page i and an ad j, we would like to predict $P_{ij} = \Pr[\text{Click}|i,j]$. The available historical data consists of matrices $C, V \in \mathbb{Z}^{m \times n}$, representing the number of *clicks* and *views* respectively that have been observed for particular (page, ad) pairs. Specifically, V_{ij} tells us how many times ad j was shown on publisher's ith page, and C_{ij} says how many times it was clicked once shown. By definition, we must have $C_{ij} \leq V_{ij}$.

Hierarchies. We use $\mathcal{H}^X$ to denote a *hierarchy* over objects in the set X. A hierarchy is a directed graph $(\mathcal{V}, \mathcal{E})$ with a *level* based structure: there is an injective function $\ell^X : \mathcal{V} \to \{1, \ldots, D^X\}$, where D^X is the *depth* of the hierarchy, such that a node $u \in \mathcal{V}$ can only have edges to nodes $v \in \mathcal{V}$ with $\ell^X(v) = \ell^X(u) + 1$. Further, every node in levels $\{1, \ldots, D^X - 1\}$ has at least one outgoing edge. The leaf nodes in level D^X correspond to the elements of X. Finally, we constrain that there is only one node $v \in \mathcal{V}$ with $\ell^X(v) = 1$, which we call the *root node*. Figure 1 represents an example of such a hierarchy over ads, where $D^X = 4$. We let $|\mathcal{H}^X| = |\mathcal{V}|$ denote the number of nodes in the hierarchy $\mathcal{H}^X$.

We refer to the set of parents for a node u by $\text{Par}(u)$. A hierarchy is a *tree* if and only if every non-root node has exactly one parent. Any two nodes u, u' with a common parent are called *siblings*. We refer to the set of paths from the root node to a node u by $\text{Path}(u)$.

2.2 Response prediction

The introduction described the basic monetary paradigm in computational advertising, where an advertiser pays a publisher for every user click. Response prediction was cast as the problem of estimating the fundamental quantity of interest in this setting, which is the probability of an ad being clicked. Recently, new response types have been adopted by advertisers, where the payout happens not when a user clicks on an ad, but only when she performs an action (called a *conversion*) after the ad is displayed. The action could be buying a product, filling out a form, *et cetera*. Accurate estimation of the probability for all response types is crucial to select the ad with highest expected revenue. "Response prediction" is therefore a blanket term that includes prediction for all three types of user response. However, the specific type of conversion does not change the modelling problem, and so for the rest of the paper, we will assume we are dealing with click events for simplicity.

With this is mind, consider the problem of predicting $P_{ij} = \Pr[\text{Click}|\text{View}; i, j]$. Given historical data, arguably the simplest solution is the maximum likelihood estimate (MLE), P^{MLE}:

$$P_{ij}^{\text{MLE}} := \begin{cases} \frac{C_{ij}}{V_{ij}} & \text{if } V_{ij} \neq 0 \\ ? & \text{else.} \end{cases} \quad (1)$$

The "?" denotes that the MLE is undefined when $V_{ij} = 0$, since we have no historical data for the particular (page, ad) pair. Thus, the MLE is unusable if $V_{ij} = 0$. Similarly, if $V_{ij} \neq 0$ but is small, then the MLE is very noisy: for example, if an ad has been shown 5 times on a page and received 0 clicks, a CTR estimate of 0 is intuitively too extreme. As these two cases are the majority in practice, we need a different estimate $\hat{P}$ that *smoothens* the MLE to make it more reliable. There are three properties any estimator $\hat{P}$ should satisfy: (i) it should provide estimates when $V_{ij} = 0$, (ii) it should provide smoothened estimates when $V_{ij} \neq 0$ but is small, and (iii) it should output meaningful probabilities in $[0, 1]$. Note however that the MLE is *consistent*, meaning that as $V_{ij} \to \infty$, $P_{ij}^{\text{MLE}} \to P_{ij}$. This implies that any estimator $\hat{P}_{ij}$ should ideally *converge to the MLE when V_{ij} is sufficiently large*.

Existing learning methods for response prediction can be categorized as feature-based or MLE-based. Feature-based methods build prediction models based on explicit features of an ad and a page, also known as *side-information*. These could include the textual content of an ad, its placement on the webpage, *et cetera*. Many existing feature based methods build prediction models using the logistic regression family [17, 8]. MLE-based methods smoothen the raw MLE via statistical models of clicks and views, with a popular choice being the Gamma-Poisson model [5, 2].

2.3 Hierarchical constraints

Pages and ads can often be organized into pre-defined hierarchies. For example, the set of all ads can be categorized according to the advertiser who made the ad. Each advertiser in turn typically instruments a series of campaigns in which to display the ads, each with differing markets and goals. This means that there is a tree-like structure to the ads: Root → Advertiser → Campaign → Advertisement. See Figure 1 for a simple example of an ad hierarchy. Notice that hierarchies are not necessarily trees, since an ad may be shown as part of multiple campaigns, for example.

Hierarchical structure encodes useful prior knowledge about CTRs. For example, the CTRs of all ads from the same campaign tend to correlated with one another. So, if a particular (page, ad) pair has

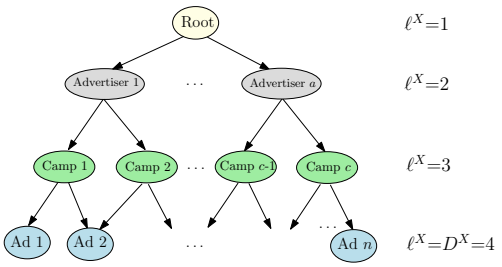


Figure 1: Hierarchical structure for advertisers.

only a few views, but the siblings of the ad have many views, then the reliability of its estimate can be increased by "borrowing" from the siblings' probability estimates in some principled manner. Hierarchical information has been successfully used in previous work on response prediction. [3] enforces relationships between estimates for (page, ad) pairs using a Markov model based on the hierarchy. The state-of-the-art LMMH model in [2] stores a separate weight ϕ_{ij} for each pair of nodes (i, j) in the page and ad hierarchies, and the click probability is modelled as a product of weights for all pairs of nodes on the respective hierarchy paths for i and j. The key idea here is *fallback*: if a (page, ad) pair (i, j) has an insufficient number of views, then the model sets ϕ_{ij} to 1, causing a fall back onto the probability estimates of parent nodes.

2.4 Collaborative filtering

Given a database of users and their preferences for various items, recommender systems aim to suggest to users new items that they will probably like, but have not already consumed. A canonical example is movie recommendation, a problem popularized by the Netflix challenge [1]. Collaborative filtering is a very successful approach to this problem, which makes recommendations based solely on the preference database: the fundamental intuition is that "similar" users will have "similar" preference, where "similar" refers to a latent notion that is *learned* by the model.

Formally, our input is an incomplete matrix $X \in \mathbb{R}^{m\times n}$ of user-item preferences, where most X_{ij} entries are missing. Our goal is to fill in these missing entries with predicted scores. The state of the art in collaborative filtering is *matrix factorization*: X is modelled as $X \approx \alpha^T\beta$, where $\alpha \in \mathbb{R}^{k\times m}$ and $\beta \in \mathbb{R}^{k\times n}$ and k is the number of *latent features* [12]. Conceptually, each α_i represents a feature vector for users, and each β_j a feature vector for items. The simplest factorization model is to solve the optimization [19]

$$\min_{\alpha,\beta} \frac{1}{|\mathcal{O}|} \sum_{(i,j)\in\mathcal{O}} (X_{ij} - \alpha_i^T\beta_j)^2 + \Omega(\alpha,\beta) \qquad (2)$$

where $\Omega(\alpha,\beta) = \frac{\lambda_\alpha}{2}||\alpha||_F^2 + \frac{\lambda_\beta}{2}||\beta||_F^2$. The first term makes the model parameters predictive of the observed entries. The second term is a *regularizer* that prevents overfitting on the observed entries. Penalizing the α and β weights by their ℓ_2 norm is equivalent to imposing a Gaussian prior on them [9, p. 64].

2.5 Our contributions

There are two main contributions to this paper. First, we present a novel approach to response prediction based on matrix factorization techniques from collaborative filtering. This model uses both latent features as well as side-information in the form of page and ad features. Ours is not a mere black-box application of collaborative filtering, however: as we will discuss subsequently, matrix factorization models such as Equation 2 are inadequate for response prediction, and require fundamental modifications. We additionally propose an iterative procedure to leverage the complementary nature of latent and explicit features. To our knowledge, the only prior work on using collaborative filtering for response prediction is [6], but it uses mixture-model approaches that have been superseded by matrix factorization. While [6] concludes that these mixture-models do not deal with the rare event problem in response prediction data, our model demonstrates state-of-the-art performance.

Second, we provide a principled way to incorporate hierarchies into matrix factorization methods for collaborative filtering. The modelling details here are very different from how hierarchies are used in existing response prediction models, since our approach is based on a different foundation. There has been work on exploiting hierarchical information in the CF literature, but only using neighbourhood models [22, 23], and so the modelling details are again completely different. Note also that such approaches are not applicable to response prediction for the same reasons that apply to standard factorization methods, which we discuss in Section 3.

3. RESPONSE PREDICTION VIA MATRIX FACTORIZATION

We now discuss how response prediction can be solved using ideas from collaborative filtering (CF). We start by addressing why a black-box application of CF techniques is insufficient.

3.1 Differences between CF and response prediction

Recall that the goal in response prediction is to construct an estimator $\hat{P}$ of the CTR that satisfies the three requirements outlined in Section 2.2. If we treat the matrix P^{MLE} as our input, then requirement (i) asks us to fill in the missing entries of this matrix. We observe an immediate connection between this task and CF: if we treat pages as users, ads as items, and the entries of P_{ij}^{MLE} as being an "affinity" score of the (page, ad) pair (i, j), then we are trying to predict the affinity for (page, ad) pairs for which there is no historical data. Applying CF to solve this problem is based on the intuition that "similar" ads will demonstrate "similar" CTRs on a given page, where "similar" is some notion that is *learned* by a mathematical model.

With this connection in place, it might appear that response prediction can be solved by just feeding P^{MLE} as the input to a collaborative filtering model, such as matrix factorization. However, a standard matrix factorization $P^{\text{MLE}} \approx \alpha^T\beta$ as in Equation 2 does *not* meet requirements (ii) and (iii): it does not output valid probabilities in $[0, 1]$, and it completely ignores the issue of confidence for entries with a few views. This necessitates fundamental changes to the factorization framework, which we detail below.

3.2 Confidence-weighted factorization

We can think of the problem of estimating $\Pr[\text{Click}|\text{View}; i, j]$ as an analogue of the binary classification problem of estimating $\Pr[y = 1|x]$. If we consider a click to be a "positive" event, and a non-click a "negative" event, we can think of the dyad (i, j) as comprising C_{ij} *duplicated* positive labels, and $V_{ij} - C_{ij}$ duplicated negative labels. Therefore, we have constructed a two dimensional table, where each cell consists of several binary labels. [13] constructs a similar table, but unlike our setting uses it in conjunction with explicit features for pages and ads.

There are now two problems to solve: we need to compute the probability of each individual label in the table being positive or negative, and we need to deal with the fact that there are multiple such labels per cell. For the first problem, suppose for the moment

that each cell (i, j) has a single binary label X_{ij}. We now assume that for a fixed set of page latent vectors α, the probability of a positive label arising for ad j may be modelled via logistic regression with a weight vector β_j, giving us $P_{ij}^{\text{MF}} = \sigma(\alpha_i^T \beta_j)$. This is clearly symmetric to assuming that for a fixed set of ad vectors β, the probability of a positive label for a page i has the given form. If we now optimize over α,β jointly, we get the modified factorization model

$$\min_{\alpha,\beta} \frac{1}{|\mathcal{O}|} \sum_{(i,j)\in\mathcal{O}} \big(-X_{ij} \log P_{ij}^{\text{MF}}(\alpha,\beta) - (1_{ij} - X_{ij}) \log(1 - P_{ij}^{\text{MF}}(\alpha,\beta))\big) + \Omega(\alpha,\beta), \quad (3)$$

where $\Omega(\alpha,\beta) = \frac{\lambda_\alpha}{2}||\alpha||_F^2 + \frac{\lambda_\beta}{2}||\beta||_F^2$ as in Equation 2. The above objective can be thought of as replacing the square-loss in Equation 2 with the more appropriate logistic loss for binary data. Models of this type have been recently studied in the collaborative filtering literature, such as in the RLFM model [4], which optimizes parameters in a Bayesian manner. In statistics, similar models have been considered in the context of modelling edges in graphs [10]. SGD optimization of the above has been proposed in the LFL model [15], of which the above is a special case for binary inputs, and the FIP model [21]. Finally, [18] proposed a similar model for collaborative filtering, where P_{ij}^{MF} was further scaled to constrain predictions to lie in a finite interval.

In the above, each α_i represents a latent feature vector for pages, and each β_j a latent vector for ads. This assumes a *low rank* structure to P, which induces correlations between the matrix entries. This sharing of parameters allows us to smoothen probability estimates: even if a particular (page, ad) pair (i, j) has a few views, we can still reliably estimate $\hat{P}_{ij}$ by estimating α_i from all other ads shown on the page, and β_j from all other pages the ad is shown on.

The remaining problem is how to deal with the multiple entries in each cell of the table. Noting the connection of the above model to logistic regression, the duplicated labels in the table are analogous to having the same example x appear several times in a supervised learning task, each time with a different labels. Logistic regression can easily be performed on such a dataset, and it is straightforward to check that the number of duplicated labels merely appear as weights in the likelihood term. The same holds in our problem, and so we get the objective

$$F(\alpha,\beta\,; C, V) = \frac{1}{|\mathcal{O}|} \sum_{(i,j)\in\mathcal{O}} \big(-C_{ij} \log P_{ij}^{\text{MF}} - (V_{ij} - C_{ij}) \log(1 - P_{ij}^{\text{MF}})\big) + \Omega(\alpha,\beta). \quad (4)$$

where P^{MF} is understood to be a function of α,β. This objective clearly takes into account confidence in the entries: if a dyad has small V_{ij}, then very little weight is placed on the likelihood term that measures its probability of click. The form of the objective is somewhat similar to that of [11], which targets CF datasets where there are no negative ratings. However, that paper uses a factorization of the form $X \approx \alpha^T \beta$, which as mentioned earlier does not output valid probabilities. Further, our objective is motivated by a principled construction of a table of positive and negative labels from the C, V matrices.

The objective in Equation 4 is differentiable, and may be optimized using stochastic gradient descent (SGD). This allows us to scale to very large datasets. The variations to the basic objective that we describe in subsequent sections will not affect differentiability, and so they will all be amenable to SGD training. As with the standard objective in Equation 2, the objective in Equation 4 is non-convex and thus SGD optimization only finds a local minima. Empirically, this local minima is found to have good predictive performance on test data.

3.3 Is factorization better than other methods?

Our confidence-weighted factorization overcomes the limitations of MLE. First, it can make predictions for (page, ad) pairs with no historical views. Second, even when there are historical views, it returns a *smoothened* estimate that is learned from multiple (page, ad) pairs, which has lesser variance than the raw MLE value. One caveat is that our method is *not* guaranteed to converge to the MLE as $V_{ij} \to \infty$, simply because the true probability P_{ij} may not be expressible as $\sigma(\alpha_i^T \beta_j)$. However, since we noted that our optimization can be thought of as using the logistic loss, which is a *proper loss* function [7], we can expect our estimates to be meaningful approximations of P_{ij}. In Section 6 we will demonstrate that empirically, given a large enough number of latent features k, our model converges to the MLE solution when V_{ij} is large.

Recall from Section 2.2 that existing methods for response prediction are based on either explicit features or MLE smoothing. We will see what potential advantages the other methods have over the factorization, and use these to guide extensions to our model. Looking at the feature-based methods, the obvious difference to our basic factorization model is that we learn *latent* features from the historical data, and use these to make predictions. In standard collaborative filtering problems, latent features are typically much more expressive than explicit features, because they can capture subtle correlations in the data [16]. However, the features used in response prediction – for example, the spatial placement of the ad on a webpage, the time it was displayed, *et cetera* – are known to be highly influential to CTR by themselves, and so a similar conclusion cannot obviously be made here. Instead, the approaches should be seen as *orthogonal* solutions that should be combined.

Compared to simple statistical models for smoothening the MLE, one advantage of the factorization approach is that we can learn a rich latent structure by choosing a sufficiently large number of latent features k. However, the state-of-the-art LMMH technique [2] warrants a closer study. LMMH is based on two ingredients. First, the results of a baseline model only using features (such as logistic regression) are used to compute an expected number of clicks $\hat{C}_{ij}$ for each dyad. Next, one fits a log-linear model on the clicks C and *expected* clicks $\hat{C}$ under some statistical assumptions about the data. This log-linear model is fitted by exploiting *hierarchical* information for the pages and ads. This information is treated separately from the other features (such as placement of an ad) because of its special structure: specifically, as we noted in Section 2.3, the hierarchy for an ad directly encodes information about correlations amongst CTRs. The reason is that this is a high-dimensional categorical variable, whose outcomes appear amongst multiple cells in the data matrix. The use of hierarchies helps LMMH handle the extreme sparsity problem quite effectively.

With this understanding, we now study two important extensions to the factorization model in turn: how to incorporate side-information, and how to incorporate hierarchies. The resulting model will be shown to have superior performance to both LMMH and the feature-based methods. Table 1 provides a summary of the components of our final model.

4. INCORPORATING SIDE-INFORMATION

As discussed earlier, pages and ads possess explicit features other than just their unique identifiers, such as the content of the ad, its placement on the page, *et cetera*. In collaborative filtering, such features are known as *side-information*. Side-information is primarily useful for making predictions in *cold-start* settings, where

Method	Section	Key idea
Confidence-weighted factorization	Section 3.2	$P_{ij}^{\text{MF}} \approx \sigma(\alpha_i^T \beta_j)$ with confidence determined by C_{ij}, V_{ij}
Fusion with side-information	Section 4.1	$P_{ij}^{\text{SI}} \approx \sigma(\alpha_i^T \beta_j + w^T x_{ij})$ for side-information given in x_{ij}
Iterative refinement	Section 4.2	$P_{ij}^{\text{MLE}} \rightarrow P_{ij}^{\text{SI}}$ and confidence weighted factorization repeated
Hierarchy regularization	Section 5.1	$\alpha_i \sim \mathcal{N}(\alpha_{\text{Par}(i)}, \lambda_\alpha^{-1} I), \beta_j \sim \mathcal{N}(\beta_{\text{Par}(j)}, \lambda_\beta^{-1} I)$
Agglomerate fitting	Section 5.2	$\alpha_i \sim \mathcal{N}(\alpha_{\text{Par}(i)}, \lambda_\alpha^{-1} I), \beta_j \sim \mathcal{N}(\beta_{\text{Par}(j)}, \lambda_\beta^{-1} I)$; $C, V \rightarrow C^{\text{Agg}}, V^{\text{Agg}}$
Residual fitting	Section 5.3	$\alpha_i \rightarrow \sum_{u \in \text{Path}(i)} \alpha_u, \beta_j \rightarrow \sum_{v \in \text{Path}(j)} \beta_u$
Hybrid hierarchical model	Section 5.4	Hierarchy regularization + Agglomerate fitting + Residual fitting

Table 1: Summary of the various components of the model used in this paper.

we have a dyad involving an object that was unobserved during training; for example, we may have an ad that has never been displayed before. In response prediction, the explicit features are strongly predictive of the CTR, as demonstrated by the fact that methods based just on explicit features were until recently the state-of-the-art [17]. It is thus prudent to incorporate this information into the factorization model for improved accuracy.

4.1 A joint factorization and feature model

Several ways of combining latent and explicit features have been studied in the collaborative filtering literature. We will use a simple scheme employed by the LFL [15] and FIP models [21], where we have a linear combination of the latent features and explicit features:

$$P_{ij}^{\text{MF}} = \sigma(\alpha_i^T \beta_j + w^T x_{ij}),$$

where x_{ij} consists of the explicit features for the dyad (i, j). It is clear that our predictions will now be influenced by the side-information, with w measuring the relative importance of the explicit features over the latent features. One now optimizes for the latent vectors α,β as well as the weights w. Training the augmented model can be done in an alternating fashion as suggested in [15]. We describe a simplified version of this process: first, note that the model may be rewritten as

$$P_{ij}^{\text{MF}} = \sigma(\left[1; w\right]^T \left[\alpha_i^T \beta_j; x_{ij}\right]), \tag{5}$$

This can be seen as a logistic regression model where the matrix factorization estimates $\alpha_i^T \beta_j$ are treated as additional *input features* that are augmented with x_{ij}. This suggests a simple learning strategy: first, we train the standard factorization model, yielding estimates P^{MF}. This step is equivalent to assuming that $w \equiv 0$. Now we fix $\alpha_i^T \beta_j$, and just optimize over w. This can be done by feeding the features $x'_{ij} = \left[\alpha_i^T \beta_j; x_{ij}\right]$ into a standard logistic regression model. The resulting solution, P^{SI}, predicts the click probability using both latent structure as well as side-information, thus enjoying the benefits of both approaches.

4.2 An iterative refinement procedure

Given the above model, a simple iterative procedure can be applied to further improve performance. Let us rewrite the confidence weighted factorization model of Equation 4 as

$$\min_{\alpha,\beta} \frac{1}{|\mathcal{O}|} \sum_{(i,j)\in\mathcal{O}} -V_{ij} \left(P_{ij}^{\text{MLE}} \log P_{ij}^{\text{MF}} - (1 - P_{ij}^{\text{MLE}}) \log(1 - P_{ij}^{\text{MF}})\right) + \Omega(\alpha,\beta). \tag{6}$$

Earlier, we motivated the above as a sensible way to incorporate confidence into the factorization model. However, the model is limited by the historical data: recall that if a cell has a small to medium number of views, then P_{ij}^{MLE} will be noisy. This means that our confidence weighting is *itself* noisy. Ideally, we would like to weight each entry by the true probability P_{ij}, which is the optimal measure of confidence to guide our factorization model. But of course, if we knew this quantity, our learning would be complete. Yet this motivates the following EM-style procedure: we take the predictions from our above model, P_{ij}^{SI}, and use these in place of P_{ij}^{MLE} in the above equation. We now *re-learn* our confidence-weighted factorization model using these new confidences. The idea is that since P_{ij}^{SI} is a more reliable estimate of the true probability that P_{ij}^{MLE}, the factorization model is encouraged to focus its modelling efforts on the "right" dyads. We can now iterate by feeding the results of the newly learned factorization into a logistic regression model, and use the resulting estimates $\hat{P}_{ij}^{SI}$ as a fresh set of confidence weights. This process may be repeated till convergence. This scheme exploits the complementary properties of the two models so that by improving the outputs of one model, we can feed in more reliable inputs to the other.

We note a similarity to the previously discussed approach followed in LMMH, with two subtle but important differences: (i) LMMH uses a feature-based model to generate the initial refinement of the data, based on which further smoothing is applied. By contrast, we apply the *factorization* as the first step to refine the data, and feed this into the feature-based model; and (ii) We iterate the process till convergence. Distinction (ii) can naturally be expected to improve performance, a fact we verify empirically. We argue that distinction (i) is important, because our factorization approach yields much more reliable estimates than a feature-based method if we have even a small amount of historical data, as observed in [16]. The smoothing applied in the first phase fundamentally affects the overall performance, because only if the results of the first model are sufficiently rich does the second stage phase gain a significant advantage over modelling the training data as-is. Put another way, LMMH is made to solve the difficult problem of fitting the residual of an only moderately reliable feature-based model. By contrast, in our case the second phase involves a much simpler task, since most of the structure is already captured by the factorization. These intuitions will be empirically corroborated in Section 6.

5. INCORPORATING HIERARCHIES

Recall that the other ingredient behind the success of existing response prediction methods is the use of hierarchical information to overcome the extreme data sparsity. We now look how this can be incorporated into our factorization model. There are three basic ideas: (i) *hierarchical regularization*, where we keep a latent vector for each node in the page and ad hierarchies, and enforce priors based on this to induce correlations; (ii) *agglomerate fitting*, where we refine (i) by constructing more informative priors based on agglomerating subtrees in the hierarchy; and (iii) *residual fitting*, where we use fit the residual of the basic factorization using

appropriate latent vectors in the hierarchy. Each of the methods modifies different parts of the objective function in Equation 4 (as noted earlier, without altering differentiability). The methods can in fact be used in *conjunction* with each other, and thus represent different facets of a larger approach to incorporating hierarchies. We will return to this matter after first discussing the methods in detail.

5.1 Hierarchical regularization

Our first idea is to let every node in the hierarchy possess its *own* latent vector, and use this to construct priors that constrain the latent vectors. For simplicity, we will use the notation α_i to denote the appropriate vector associated with the node i, and similarly for β_j: in Figure 2, $\beta_1, \ldots, \beta_n$ represent the latent vectors for ads as before, while $\beta_{n+1}, \ldots, \beta_{n+c}$ represent the latent vectors for the campaigns, and so on. When we refer to the matrix α, it is now in $\mathbb{R}^{|\mathcal{H}^P| \times k}$, and similarly for β.

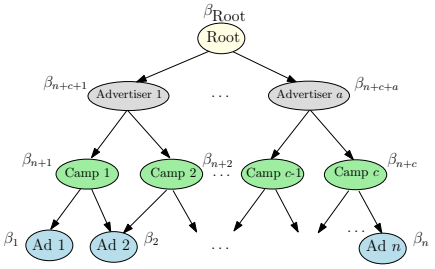


Figure 2: Each node in the hierarchy has a latent vector.

Recall that the standard ℓ_2 regularizer on α,β corresponds to placing a zero-mean Gaussian prior on them, i.e. $\alpha_i \sim \mathcal{N}(0, \lambda_\alpha^{-1} I)$. With the above setup, we impose *hierarchical priors* on the latent vectors, such that each latent vector has a prior so that it behaves like its parent in expectation:

$$\alpha_{\text{Root}} \sim \mathcal{N}(0, \lambda_{\text{Root}}^{-1} I), \forall i \in \mathcal{H}^P - \{\text{Root}\} \alpha_i \sim \mathcal{N}(\alpha_{\text{Par}(i)}, \lambda_\alpha^{-1} I) \tag{7}$$

To find the joint prior over α, note that given its parent, a node is conditionally independent of all higher level nodes. Thus, we replace the standard regularizer in Equation 4 by

$$\Omega(\alpha,\beta) = \frac{\lambda_\alpha}{2} \sum_{i \in \mathcal{H}^P} ||\alpha_i - \alpha_{\text{Par}(i)}||_2^2 + \frac{\lambda_\beta}{2} \sum_{j \in \mathcal{H}^A} ||\beta_j - \beta_{\text{Par}(j)}||_2^2 + \frac{\lambda_R}{2}(||\alpha_{\text{Root}}||_2^2 + ||\beta_{\text{Root}}||_2^2). \tag{8}$$

The regularizer helps in estimating vectors when there are few corresponding views. Suppose there are two siblings u, v with a common parent, and that node u has only a few views while node v has many views. For v, the dominating term in the objective will be the loss function, so its parameters will be optimized to be predictive for the CTR. For u, the regularizer will dominate and push its latent vector to be similar to the parent node. In turn, the parent is encouraged to be close to its children, and so u, v are indirectly encouraged to be similar to each other. This means that u will "borrow strength" from v.

The above easily handles hierarchies that are not trees: instead of the prior mean being a parent node's latent vector, we can use the *average* of all parents' vectors. This approach has a total of $(|\mathcal{H}^P| + |\mathcal{H}^A|)k$ parameters. In practice, hierarchies are often bottom-heavy, so that $|\mathcal{H}^X| = O(|X|)$. Thus we learn roughly the same number of parameters as in the standard setting. Of course, the constant in the $O(\cdot)$ makes a difference in two ways: it adds the risk of overfitting, and it potentially increases the number of local optima. We will empirically verify that neither of these risks are seriously manifested in practice.

5.2 Agglomerate fitting

In the previous section, the latent vectors for non-leaf nodes only appear in the regularizer, and hence are only indirectly affected by the click and view data. Intuitively, by making them directly depend on the data, they will serve as more informative priors for the leaf nodes. To do this, we use the hierarchy to *agglomerate* the click/view data across many pages and ads, and try to predict this data using the appropriate latent vectors. For example, for a (page, campaign) pair (i, c), we agglomerate the clicks/views for all children of c when shown on page i. We model the resulting data using the vectors α_i and β_c. This will learn a sensible prior for the children's latent vectors.

Formally, we construct the Cartesian product of the hierarchies, $G := \mathcal{H}^P \times \mathcal{H}^A$. Any $(u, v) \in G$ is a tuple of nodes from the page and ad hierarchies, and we associate with it the aggregated clicks of all its children:

$$C^{\text{Agg}}(u, v) = \sum_{(u',v'):(u,v) \in \text{Par}((u',v'))} C^{\text{Agg}}(u', v').$$

The base case is when both u and v are leaf nodes in the respective hierarchies, so that $C^{\text{Agg}}(u, v)$ is the standard click value C_{ij}. We repeat the same process for the views. We now have click and view matrices $C^{\text{Agg}}, V^{\text{Agg}} \in \mathbb{R}^{|\mathcal{H}^P| \times |\mathcal{H}^A|}$. We optimize the objective of Equation 4 using $F(\alpha,\beta\,; C^{\text{Agg}}, V^{\text{Agg}})$ along with the regularizer Ω of Equation 8. Therefore, every latent vector appears in both the objective and the regularizer. Figure 3 illustrates how we can think of the new matrices as being augmentations of the original C, V.

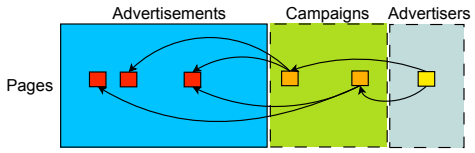


Figure 3: Illustration of the agglomeration process. Arrows denote that the clicks/views are added up.

There is a subtle problem with implementing agglomerative fitting as-is: the clicks and views of individual (page, ad) pairs will be "drowned out" by the ones corresponding to parent nodes, and our predictions will no longer be fine-grained for each individual pair. Concretely, consider some page i, an ad j, and the ad's campaign c. Then, the number of clicks/views for the (page, campaign) pair (i, c) will be greater than the clicks/views for the (page, ad) pair (i, j), by construction of the agglomerated click and view matrices. Since the model of Equation 4 is confidence weighted by the number of views, we will essentially ignore the contributions of the (page, ad) pairs. One fix to this problem is to train our model in stages. Letting α^{Par} and β^{Par} denote the latent vectors for non-leaf nodes, we perform the following steps:

1. Learn α,β with $\alpha^{\text{Par}}, \beta^{\text{Par}}$ fixed. This is the standard factorization model of Equation 4.
2. Learn $\alpha^{\text{Par}}, \beta^{\text{Par}}$ with α,β fixed. This asks the parent nodes to be predictive for the agglomerated data, but *using* the already learnt vectors for pages and ads e.g. when we predict for a (page, campaign) pair.

3. Relearn α,β with α^{Par}, β^{Par} fixed. This asks us to respect the hierarchical prior when reconstructing the data matrix. Since the parent nodes are not optimized at this stage, we do not overfit on the agglomerated entries. We can use the result of step (1) as initialization here.

One can think of this process as using hierarchies to give a principled way of finding good local optima for α,β .

5.3 Residual fitting

The previous extensions all enforce similarity among parameters based on the hierarchy, but finally use the prediction $\sigma(\alpha_i^T \beta_j)$. A sensible idea is to modify this prediction itself based on the hierarchy. Specifically, for the pair (i, j), we use the prediction $\sigma(\tilde{\alpha}_i^T \tilde{\beta}_j)$, where $\tilde{\alpha}$, $\tilde{\beta}$ modify the original vectors α,β based on the hierarchy. A simple choice is the additive model

$$\tilde{\alpha}_{ik} = \alpha_{ik} + \sum_{u \in \text{Path}(i) - \{i\}} \alpha_{uk}$$

and similarly for $\tilde{\beta}$. Here, the fine-grained latent features for each page are modelled as corrections over coarser latent features of parent nodes, which may be thought of as *bias* terms. This technique easily generalizes to non-tree hierarchies if we use the average latent vector of all parent nodes on a *per-level* basis.

5.4 Putting it all together: a hybrid method

We can easily combine the ideas of the previous sections to create the following hybrid approach: (i) our prediction for dyad (i, j) involves all latent vectors along the respective paths in the hierarchy, (ii) we impose hierarchical priors on the latent vectors, and (iii) we force the latent vectors for parent nodes to be predictive for agglomerated data. We later demonstrate that empirically, this hybrid performs significantly better than any of its individual components. We also note that the hybrid can be easily augmented with additional side-information using the framework detailed in Section 4.

5.5 Handling cold-start pages and ads

We quickly comment on how the hierarchical extensions let us estimate the latent vectors for a cold-start page or ad. As a simple example, suppose there are several pages that share a common parent node. Suppose one of them, page i, has no past history. Then, if we use hierarchical regularization, α_i will only appear in the regularization term Ω in Equation 8. Thus, it optimized by setting it equal to $\alpha_{\text{Par}(i)}$. If this parent node is estimated via agglomerative fitting, it will be a good representative of the common structure between the siblings of i. Therefore, we will be able to make reasonable predictions for page i on test data. This argument holds equally for entities that are "almost cold-start" i.e. which have extremely limited past history. Having inferred the latent vectors in this manner, we can now employ the framework of Section 4.

6. EXPERIMENTS

Our experiments address several questions: (i) how well does the basic confidence-weighted factorization model perform compared to existing response prediction methods?, (ii) do the hierarchical extensions improve performance?, and (iii) does exploiting latent features and side-information offer noticeable performance gains? To answer these questions, we conducted experiments on three very large real-world datasets, which shows our model is highly scalable and that it outperforms current state-of-the-art methods. MATLAB code for our methods is available online at [14].

6.1 Datasets used

Our datasets were collected from Yahoo! traffic streams. (Unfortunately, we cannot release our datasets due to their proprietary nature.) To mirror the real-world prediction problem, the train-test splits in all our datasets is done in a temporal manner, so that the events in the training set occur before those in the test set. All splits were conducted 20 times to ensure the significance of our results. Since the datasets are proprietary, we cannot report the corresponding number of pages and ads, clicks and views, nor the number of days used to construct the train-test split. Instead, we just report the number of nonzero records (≥ 1 view) in the train and test sets.

We used the same three datasets as [2]. These datasets each involve the interaction between ads and pages, and differ in the nature of the interaction. The first data set, **Click**, records the event of an ad being clicked when shown on a page. Both pages and ads have a two level hierarchy in this dataset. The second dataset, **PVC**, measures how many users performed a pre-determined action after *viewing* an ad. Finally, the post-click conversion dataset (**PCC**) measures how many users performed a pre-determined action after *clicking* an ad. Both **PVC** and **PCC** have four level ad hierarchies, and the same two level page hierarchy as with **Click**. Of the three datasets, **PVC** is the sparsest, since post-view conversions are generally difficult to measure. There are $\sim$(90B, 3B) (train, test) records for **Click**, $\sim$(7B, 250M) for **PVC**, and $\sim$(500M, 20M) for **PCC**. The three datasets also include *interaction features* for the user involved in each interaction (e.g. the age and gender of the user that clicks on an ad, how recently the ad was shown to the user, *et cetera*).

6.2 Methods used

We implemented the confidence-weighted factorization of Section 3 (denoted "CWFact" in our figures), two of our hierarchical extensions (denoted "Agglomerative", and "Residual"), and the hybrid hierarchical method of Section 5.4 (denoted "Hybrid"). (The hierarchical regularization method in Section 5.1 by itself showed similar performance to CWFact, as is essentially subsumed by Agglomerative.) The basic versions of these methods did not include explicit features, and are purely based on matrix factorization. We additionally ran the CWFact and Hybrid methods with side-information using the joint model of Section 4, and denote these methods by "CWFact+LogReg" and "Hybrid+LogReg". Finally, the method "Hybrid+LogReg++" used the iterative scheme detailed in Section 4.2, where we iteratively use the model's predicted number of clicks as input to a residual model. We ran this model till convergence.

We trained all models using stochastic gradient descent (SGD), and used the MapReduce code from the Apache Mahout project (`http://mahout.apache.org/`) to scale to the challenging sizes of the datasets. We parallelized the optimization by fixing the page latent features α_i and then optimizing for the advertisement latent features β_j using SGD. This optimization of each individual β_j can be done in parallel. Once the estimates of the β_j's converged, we fixed their values and optimized the α_i's in parallel; this alternating scheme was repeated until convergence. For the hierarchical regularization scheme, we optimized the parameters level-by-level based on the hierarchy. Thus, for the initial optimization of advertisement features β_j, the campaign features were fixed and regularization was done towards these fixed latent features. Once optimization of the advertisements concluded, the campaign features were optimized with regularization towards fixed advertiser features, and so on.

Regarding parameter selection, we picked the strengths of regularization λ_α, λ_β by cross-validation. Due to space limitations, we

cannot show results for a range of latent feature dimensions, k, and instead just show results for $k = 100$. Qualitatively, we found that the gains of our model were modest with a much smaller value of $k = 10$, indicating that it is important to choose a sufficiently large number of latent features to capture the structure in the data.

We compare to three state-of-the-art methods, discussed earlier in Section 2.2. The first is the model of [2], denoted "LMMH". The second and third are variants of a logistic regression model that uses explicit features for pages and ads, similar to [17]. Prior to LMMH, these models were the state-of-the-art for response prediction [17]. We fed these logistic methods input features derived from the page and ad hierarchies. For example, each ad has its corresponding advertiser ID as a categorical feature. The first variant of logistic regression, "LogReg", just used all the raw features as input. The second variant, "LogRegHash", also uses *cross-features* of pages and ads. To mitigate the dramatic increases in the number of features, following [2], we applied the hashing trick of [20] to compress the features into a smaller number of bins.

6.3 Evaluating performance

We use the standard performance metric of Bernoulli log-likelihood, which measures how well the predicted CTRs match the CTR on the test set. For a test set $\mathcal{T}$ of (page, ad) pairs with clicks and views $C^{\mathcal{T}}, V^{\mathcal{T}}$, given the model's predictions $\hat{P}_{ij}$, the log-likelihood is:

$$\mathcal{L} = -\sum_{(i,j)\in\mathcal{T}} \log \hat{P}_{ij}^{C_{ij}^{\mathcal{T}}} (1 - \hat{P}_{ij})^{(V_{ij}^{\mathcal{T}} - C_{ij}^{\mathcal{T}})}.$$

Again, for confidentiality reasons we cannot report raw log-likelihood numbers for any method. Hence, on all datasets we report the % lift in likelihood over the appropriate baselines.

6.4 Results on large-scale datasets

Our results are summarized in Figure 4. We see that for all datasets, the Hybrid+LogReg++ model gives the best log-likelihood lift, in particular outperforming LMMH in terms of average lift, and with a consistently lower standard deviation. The multiple iterations used in Hybrid+LogReg++ are seen to have a useful boost over Hybrid+LogReg, which by itself outperforms LMMH on all but PCC, where LMMH performs marginally better. This shows that the combination of factorization, hierarchies and side-information gives state-of-the-art performance. Note also that the good performance of Hybrid+LogReg shows that it is not just the multiple iterations that give us the advantage over LMMH. A closer study reveals the value of each of these components in our final model. The importance of using hierarchies is demonstrated by the surprisingly poor performance of the basic CWFact model, which is outperformed by even the simple LogReg models. However, these feature-based models are in turn significantly outperformed by the Hybrid method that adds hierarchies to the factorization. Note also that the Hybrid method always manages to significantly improve over the individual Agglomerative and Residual sub-models. Yet, we conclude that it imperative to have a model that *combines* latent features and side-information, because the Hybrid method by itself is merely competitive with LMMH; it is only when we add side-information to the factorization that we start to see improvements.

For our models that used both latent features and side-information, we found that the LogReg component invariably put relatively little weight on the standard features, meaning that the factorization "feature" was considered most important. This is expected, since the factorization model by itself is strongly predictive of the CTR. To further analyze the interplay between the factorization and side-information, Figures 5a and 5b gives log-log plots of the ratio of predictions of the Hybrid+LogReg++ model and the LogReg model to the test set CTR, ordered by increasing number of views on the training set. The flat line in black is the optimal solution where the model prediction matches the test set CTR, and both are displayed on the same y-range for ease of comparison. (Recall that we are unable to report exact view numbers for our datasets. Hence, in both plots, we removed identifying information about the number of views on the x-axis.) There are two striking characteristics in the plots: first, the Hybrid+LogReg++ model has significantly less variance than LogReg, which shows that its factorization component helps the LogReg component by capturing most of the structure in the data through latent features. Second, the Hybrid+LogReg++ model converges much quicker to the true CTR than LogReg model in terms of number of views. This shows that our model can successfully smoothen at a much greater degree of sparsity in the training data, corresponding to dyads with a few number of views. As we noted earlier, similar behaviour has been observed for standard collaborative filtering data [16].

Finally, to see how the performance of Hybrid+LogReg++ varies across each iteration, Figure 5c shows the lifts on the **Click** dataset after each iteration. An iteration here refers to a single application of both the Hybrid and LogReg models, using the results of the previous iteration as input. The results are encouraging: we almost always improve the log-likelihood as we run the model for more iterations. (We observed similar results on the other datasets.) This shows that with minimal added cost, the simple iterative scheme of Section 4 offers further gains for our framework.

7. CONCLUSION AND FUTURE WORK

This paper showed how we can improve on the state-of-the-art in response prediction using a novel approach based on collaborative filtering. Our model is based on a matrix factorization that learns latent structure from the data, and additionally exploits side-information in the form of page and ad features. We proposed an iterative procedure to further exploit the complementary nature of latent and explicit features. Finally, we showed how to incorporate hierarchical information for pages and ads into our model. Experimental results on Yahoo! traffic data show that our method outperforms existing response prediction approaches, and that it can handle high amounts of sparsity in the training data.

There are several avenues for future work. First, the matrix factorization model proposed here uses MAP estimates of the weight vectors. A Bayesian approach involving marginalization of these parameters would be useful, though it poses challenges due to the interdependencies between parameters. Second, it is important to address other real-world complexities that arise in response prediction, such as the fact that ad behaviour is time-varying.

8. REFERENCES

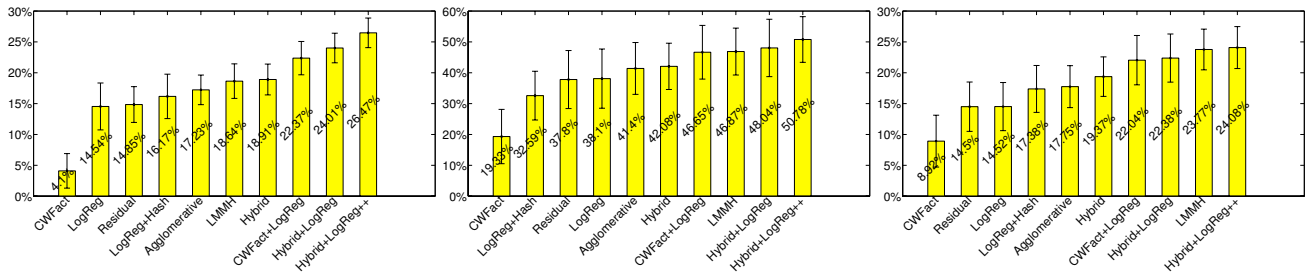
[1] Netflix prize. http://www.netflixprize.com/.

[2] D. Agarwal, R. Agrawal, R. Khanna, and N. Kota. Estimating rates of rare events with multiple hierarchies through scalable log-linear models. In *KDD '10*, pages 213–222, New York, NY, USA, 2010. ACM.

[3] D. Agarwal, A. Z. Broder, D. Chakrabarti, D. Diklic, V. Josifovski, and M. Sayyadian. Estimating rates of rare events at multiple resolutions. In *KDD '07*, pages 16–25, New York, NY, USA, 2007. ACM.

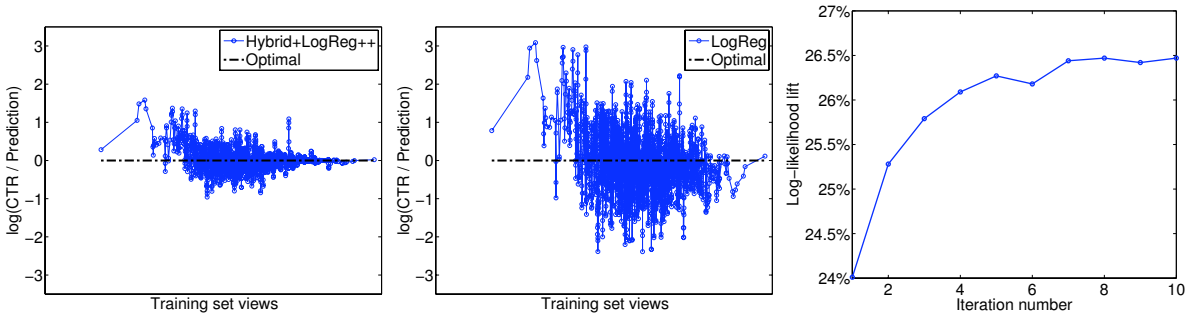
[4] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In *KDD '09*, pages 19–28, New York, NY, USA, 2009. ACM.

[5] D. Agarwal, B.-C. Chen, and P. Elango. Spatio-temporal



(a) Click. (b) PVC. (c) PCC.

Figure 4: Log-likelihood lifts on large-scale datasets.



(a) Hybrid+LogReg++ versus test CTR. (b) LogReg versus test CTR. (c) Per-iteration lift.

Figure 5: Analysis of factorization model's performance on Click. See text regarding the missing axes in (a) and (b).

models for estimating click-through rate. In *WWW '09*, pages 21–30, New York, NY, USA, 2009. ACM.

[6] S. Banerjee and K. Ramanathan. Collaborative filtering on skewed datasets. In *WWW '08*, pages 1135–1136, New York, NY, USA, 2008. ACM.

[7] A. Buja, W. Stuetzle, and Y. Shen. Loss functions for binary class probability estimation: Structure and applications. Unpublished manuscript, 2005.

[8] T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-scale Bayesian click-through rate prediction for sponsored search advertising in Microsoft's Bing search engine. In *ICML '10*, pages 13–20, 2010.

[9] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[10] P. D. Hoff. Bilinear mixed-effects models for dyadic data. *Journal of the American Statistical Association*, 100:286–295, March 2005.

[11] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM '08*, pages 263–272, Washington, DC, USA, 2008. IEEE Computer Society.

[12] Y. Koren, R. M. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

[13] Y. Lifshits and D. Nowotka. Estimation of the click volume by large scale regression analysis. In *CSR*, pages 216–226, 2007.

[14] A. K. Menon. Code for experiments. `http://cseweb.ucsd.edu/~akmenon/code`.

[15] A. K. Menon and C. Elkan. A log-linear model with latent features for dyadic prediction. In *ICDM*, 2010.

[16] I. Pilászy and D. Tikk. Recommending new movies: even a few ratings are more valuable than metadata. In *RecSys '09*, pages 93–100, New York, NY, USA, 2009. ACM.

[17] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *WWW '07*, pages 521–530, New York, NY, USA, 2007. ACM.

[18] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *NIPS*, 2007.

[19] G. Takács, I. Pilászy, B. Németh, and D. Tikk. Scalable Collaborative Filtering Approaches for Large Recommender Systems. *Journal of Machine Learning Research*, 10:623–656, 2009.

[20] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *ICML '09*, pages 1113–1120, 2009.

[21] S. Yang, B. Long, A. Smola, N. Sadagopan, Z. Zheng, and H. Zha. Like like alike – joint friendship and interest propagation in social networks. In *WWW*, 2011.

[22] C.-N. Ziegler, G. Lausen, and J. A. Konstan. On exploiting classification taxonomies in recommender systems. *AI Communications*, 21(2-3):97–125, 2008.

[23] C.-N. Ziegler, G. Lausen, and S.-T. Lars. Taxonomy-driven computation of product recommendations. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, CIKM '04, pages 406–415, New York, NY, USA, 2004. ACM.