Speeding up Large-Scale Learning with a Social Prior

Deepayan Chakrabarti
Facebook Inc.
deepay@fb.com

Ralf Herbrich
Facebook Inc.(Currently at Amazon Inc.)
herbrich@amazon.com

ABSTRACT

Slow convergence and poor initial accuracy are two problems that plague efforts to use very large feature sets in online learning. This is especially true when only a few features are "active" in any training example, and the frequency of activations of different features is skewed. We show how these problems can be mitigated if a graph of relationships between features is known. We study this problem in a fully Bayesian setting, focusing on the problem of using Facebook user-IDs as features, with the social network giving the relationship structure. Our analysis uncovers significant problems with the obvious regularizations, and motivates a two-component mixture-model "social prior" that is provably better. Empirical results on large-scale click prediction problems show that our algorithm can learn as well as the baseline with $12M$ fewer training examples, and continuously outperforms it for over $60M$ examples. On a second problem using binned features, our model outperforms the baseline even after the latter sees 5x as much data.

Categories and Subject Descriptors

I.2.6 [**Learning**]: Parameter Learning

Keywords

Social Prior, Mixture Model

1. INTRODUCTION

Large-scale online learning faces two challenges. The first is the huge number of features whose weights must be learnt. The second is sparsity and skew: any one training example typically "activates" only a few features, and the frequency of activation of various features can be heavily skewed. SVMs can deal with large (indeed, "infinite") dimensional feature spaces, but only when these features are of a very particular form (e.g., all monomials of a given degree, for polynomial kernels) and only by making computational complexity depend strongly on the number of training examples N; however, the features in large-scale learning can be arbitrary, and N is always very large. Linear models (including linear SVMs) are often the best tools in such situations, but even they converge only slowly. Due to skewed activations, learning the weights of rare features is particularly slow.

Consider, for instance, the problem of predicting clicks by users on any form of online content (e.g., news-feeds or ads) on large websites such as Yahoo! or Facebook. In addition to the "usual" features (say, user age or gender), the user's ID itself could be a binary feature — 1 if the user viewing the content has this particular ID, and 0 otherwise — potentially adding $\sim 10^9$ binary features of the form {user-ID= x} to the learning problem. The weight learnt for such a feature is then the *user-specific* bias that is left over after all other features are accounted for. Another example would be the {user-ID= x, content-category= y} binary feature, which encodes the residual propensity of user x to click on content from category y that is unexplained by other features. Each training example involves the action (click or not) of only one user when shown one particular piece of content, and hence only one of these features is activated. Clearly, learning the entire set of weights will be very slow.

However, there are many situations where features are known to be related or "close". In the example above, we may know the social network between users, and could reasonably claim that users are generally "close" to their friends and show similar biases. If so, the weights for {user-ID= u} and {user-ID= v} should be similar whenever u and v are friends. Thus, any updates to one weight (say, due to new training examples) can be propagated to other weights in its neighborhood; in effect, each training example affects multiple weights even though only one weight is activated. This should speed up the learning process, as long as our original assumption of closeness of weights was correct.

Another situation with related weights occurs when an ordinal variable (say, average time spent on website per day) is binned into non-overlapping intervals, and the user's bin becomes a binary feature {bin= x}. Normally, the weight for each such feature would be learnt independently. However, by construction, adjacent bins are related. Indeed, if the response variable (say, click or not) is a smooth function of the original ordinal variable (average time spent) and the bin-size is small enough, we expect adjacent bins to have similar weights. This insight can lead to faster learning.

While this problem exhibits superficial similarity with label propagation, the problems are actually orthogonal. Label propagation assumes that *feature values* for similar items are close, and uses this to predict missing feature values.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
KDDŠ13, August 11–14, 2013, Chicago, Illinois, USA.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
Copyright 2013 ACM 978-1-4503-2174-7/13/08 ...$15.00.

Our problem is to learn *weights* for {feature value= x}, where these weights denote how different values affect a given response variable (say, clicking behavior). The underlying assumption here is that similar feature values affect the response variable in similar way, and hence have weights that are close. Thus, weights are inherently tied to a specific prediction problem, while values are immutable (if unknown) properties of the items themselves. The same problem setting may exhibit both situations: we may need to impute missing values for some users (say, the age), *and also* the effects of these values on the click-through rate.

We formalize this similar-weights intuition into a model that places "social prior" on the feature weights. The simplest version places a $N(0, \sigma^2)$ prior on the difference between the weights of every pair of related features. While appropriate for certain situations, this formulation runs into severe problems when applied to the social network of user-IDs: in brief, some weights are forced to have tiny variances even before any training examples are observed. We analyze the reasons for this, and present a better model that provably avoids these issues.

Our contributions are as follows:

(1) We show that users in a large social network are indeed similar to their friends, in the context of clicking on ads. This effect exists even after accounting for important covariates such as the type of ad, and the context in which the ad is displayed.

(2) We propose Bayesian models that formalize this notion of feature weight similarity via a social prior. We analyze these models and show a basic problem afflicting a broad class of such models: prior variances of some weights can become too small even before any data is seen. This analysis leads us to a model that avoids this pitfall.

(3) In addition to the social network setting, we show how our proposed methods are applicable in a broad range of problems, such as when features are derived from histograms or trees. Such cascading of features — using the results of a related algorithm as input features for the learning problem — is a common technique in practice, and the proposed method can lead to faster learning in such cases as well.

(4) We empirically demonstrate the significantly faster convergence of our proposed method. For a click-prediction problem with user-ID features, the use of the Facebook social network achieves an accuracy that would otherwise require $12M$ extra training examples. These gains are primarily observed in the initial stages of learning, which is precisely where large-scale learning suffers the most. On a second real-world problem the social prior model outperforms the baseline model even when the latter sees five time as many training examples.

This paper is organized as follows. Section 2 discusses prior work. The usefulness of the social network for click-prediction is validated in Section 3. The proposed method is presented in Section 4, followed by a discussion of broader applications (Section 5) and experiments (Section 6). We conclude in Section 7.

2. RELATED WORK

We split this discussion into two parts: (a) learning algorithms using a network among features, and (b) work involving social networks in particular.

Network-based Learning. There are three major threads in this category. Network structure has been used directly via random walks with restarts in recommendation problems, and have been shown to outperform standard collaborative filtering [9]. Networks of features have also been used as regularizers, with a penalty being assessed on dissimilar weights that are connected in the network [13, 11, 15]. Regularization in a more Bayesian setting has also been proposed [5]. Finally, the network has been used as an aid in achieving sparsity of the weight vector, with a regularization term that pushes connected weights to be either both zero or both non-zero [8].

Our work is closest to the work on regularization, but our goal is to infer the weight distributions instead of optimal point-estimates. Also, there is no obvious sparsity requirement that can be used to reduce the dimensionality of the weight vector. While our first model is indeed similar to some in prior work, we show that it, in fact, is particularly poor in problems involving large social networks. We believe that our work is to first to formally analyze the reasons behind its poor performance, and to offer a solution that actually works well on a large social network and $O(10^8)$ training examples.

Utility of Social Networks. Social networks have seen recent use in a variety of problems, with varying success. It has been useful in collaborative filtering situations [9], in review quality prediction [10], and in increasing user clicks via targeted "social ads" [2]. On the other hand, Goel et al. [6] show that the social network offers less than 2% lift in CTR prediction overall, though it is more useful in other problem settings. Bao et al. [3] found that "influential" users rarely click on ads, but they can be useful for CTR prediction for *other* users, if we mine their data to find "hints" that are (a) propagated to other users, and then (b) used in the CTR estimation algorithm. Such hints can be mined separately for users and ads, and can then be used for matching, without explicitly using the social network itself [14, 17].

Thus, while social networks have been useful in general, evidence of their utility in the problem of click prediction is more ambiguous. Our focus is on using the social network to speed up convergence of the learning algorithm, which could lead to easier exploration of this topic.

3. EXPLORATORY ANALYSIS

Our underlying assumption was that the social network is informative in predicting user clicks on online content, even after accounting for the most common effects such as the type of content and the context in which the content is shown. Before we can proceed, we must justify this assumption.

If users are similar to their friends, then we should expect a user's click-through rate (CTR) to be correlated with the average CTR of her friends. However, simply calculating correlations between aggregate CTRs may not suffice, for two reasons. First, CTR can vary widely depending on aspects of the content (e.g., the content creator, such as the owner of a Facebook page for page posts, or the advertiser for an online ad) and the context in which the content was shown (e.g., desktop or mobile, news-feed or user-profile page, etc.). Hence, we compute CTR correlations only between $\text{CTR}_{\text{user}}(A, C)$ and the average $\text{CTR}_{\text{friend}}(A, C)$ for all content by content creator A shown in context C for which both the user and some friends have enough impres-

sions[1]. Second, there might be unknown biases due to the content-serving system, such as implicit connections between the number and kind of content a viewer sees and the number of friends in her social network. Hence, we need a *baseline* correlation that abstracts out just the Facebook social network while leaving other variables unchanged. We achieve this by computing correlations on randomly generated graphs where each user has the same number of friends as in the Facebook social network, but the friend-list is random [1].

We present correlations computed specifically for the case of ads shown on Facebook. Based on data collected over a short period in 2012, there were around $45M$ users who had at least 50 impressions on some (A, C) pair, and the induced subgraph had $2.3B$ friendship edges. We found a correlation of 0.273 for this subgraph, but only 0.167 ± 0.002 for the random graphs (correlations were calculated over many randomly generated graphs to get the 95% confidence intervals). The significantly higher correlations as compared to the baseline can only be due to the actual *choices* made by users in forming the social network, and it implies that ad clicking behavior of users and their friends are similar. This motivates our efforts to exploit such similarity in the click prediction problem, which we discuss next.

4. PROPOSED METHOD

Our goal here is to speed up large-scale learning in the presence of related features, under conditions of feature skew and sparsity: any single training example is likely to have only a few "active" features, and the frequency of activations of various features is likely to be very skewed over an entire training corpus. For the sake of concreteness, we shall focus on binary features of the form {value= x}, where the value could be user-ID with an associated social network, or a histogram bin for the case of a binned ordinal variable such as average time spent on Facebook, etc. In large-scale learning, the cardinality of such feature sets is by definition very large, which leads to slow learning. Sparsity limits traditional learning methods to updates of only a few features at a time, and skew implies that a large fraction of features might receive relatively little training data overall, both of which only accentuate the problem. This emphasizes the need of the solution we propose.

Our solution can be applied to *any* model linking feature weights to the response variable, as long as it uses *distributions* on the weights instead of point estimates. This is because confidence intervals on the weights can be very different due to skew in the training data, and without knowing these confidences (equivalently, variances of the weights), it is impossible to correctly propagate information between related weights, or analyze the resulting algorithm. For our experiments, we use the following model [7]:

$$p(y \mid \boldsymbol{x}, \boldsymbol{w}) = \Phi\left(y \frac{\boldsymbol{w}^t \boldsymbol{x}}{\beta}\right) \quad (1)$$

$$\boldsymbol{w} \sim \mathcal{N}\left(\mathbf{0}, \mathrm{diag}\left(\boldsymbol{\phi}^2\right)\right) \quad (2)$$

Here, y is the response variable (say, click or not), $\boldsymbol{x}$ the binary feature vector, $\boldsymbol{w}$ the weight vector, and β a known scaling factor. This is a probit model where the components w_i of the weight vector are marginally independent. Call this model NoSocial.

Now, suppose features u and v are known to be related. In the interest of clear exposition, we resume our running example of ad click prediction in the social network: u and v could refer to users who are friends (henceforth, $u \sim v$), for which we have already established a positive correlation in Section 3. This observed correlation can be incorporated into the model by positing *dependence* between w_u and w_v. Specifically, we can add in an extra "social prior" in the marginal of $\boldsymbol{w}$:

$$p(\boldsymbol{w}) = \mathcal{N}\left(\mathbf{0}, \mathrm{diag}\left(\boldsymbol{\phi}^2\right)\right) \cdot \Pi_{u \sim v} \mathcal{N}\left(w_u - w_v; 0, \sigma_{sp}^2\right) \quad (3)$$

Call this model SocialPrior. Here, σ_{sp}^2 is a constant "social prior variance"; smaller values lead to tighter coupling of weights, with $\sigma_{sp}^2 = 0$ implying identical weights for neighbors. We emphasize that this formulation can be extended to any setting where a similarity graph between features is available, and we shall discuss examples of these in Section 5.

Learning the weights. To learn the weights, we turn to a message-passing schedule: whenever a weight is updated (e.g., when new data becomes available), a corresponding message is passed to its "neighboring" weights (see [4] for an introduction on the topic). However, computation of the exact posterior is intractable, for two reasons: (1) the form of Equation 1 makes the determination of the exact message from the data y to the weights w_i difficult, and (2) the presence of loops in the social friendship graph means that message-passing need not converge, and need not yield the correct posterior variance even if it does converge [16].

The solution is to use approximations instead of the exact posterior. The first problem (which exists even in NoSocial) can be approached by using an approximation based on Expectation Propagation, as in [7]. For the second problem, we perform message-passing on a restricted set of links such that convergence is guaranteed but the results are again approximate. In particular, if user u is being shown the content, we only perform message passing on edges $u \sim v$. This set of edges is *singly-connected* so the message-passing algorithm converges, but it ignores (a) any links to users that are not direct friends of u, and (b) any links $v \sim v'$ where both v and v' are friend of u. Note, however, that every link participates in message-passing at one time or another unless both users at its' endpoints never see new content, and ignoring completely inactive users is intuitively reasonable. In this way, both problems mentioned above can be addressed by a message-passing solution that takes $O(\Delta K)$ time, where K is the maximum number of active weights for any training example, and Δ the maximum neighborhood size.

However, there is another inherent problem with this model — either the social prior variance σ_{sp}^2 must be large, or the variance of w_i must be small (in a sense that we will soon make precise). The former would imply that the components of $\boldsymbol{w}$ draw very little "strength" from their neighbors, defeating the very purpose of a social prior, while the latter renders the model inflexible even before any data is observed. Next, we state these notions precisely.

4.1 Analysis of SocialPrior

Formally, our model may be represented as a weighted graph $G = (V, E)$ where V is the set w_i, each having a standalone "prior" factor $\mathcal{N}(0, \phi_i^2)$, and $E = \{(i \sim j, \sigma_{ij}^2 | i, j \in V\}$

[1]An impression is one instance of showing the content to a user. We require at least 50 impressions of content created by A to be shown on context C for inclusion in the correlation calculation.

is a set of edges as discussed above, along with a corresponding social prior variance σ_{ij}^2 (this is a generalization of Eq. 3). This weighted graph corresponds to the graphical model of Eq. 3. As marginal inference in an arbitrary graphical model is difficult, and especially so in the presence of loops that we expect in social graphs, we develop bounds on marginals for the social prior model. Let $G = (V, E)$ be such a graphical model, and G_γ be an extension of G that increases the *precision* (i.e., inverse variance) on edge $i \sim j$: $1/\sigma_{ij}^2(G_\gamma) = 1/\sigma_{ij}^2(G) + \gamma$ (if the edge does not exist in G, it is created in G_γ). Let $p_i = p(w_i)$ be the marginal of node i (corresponding to w_i), and let σ_i^2 be the corresponding variance. Then, we have the following lemma.

LEMMA 1. *For any $k \in V$ and $\gamma > 0$, $\sigma_k^2(G_\gamma) < \sigma_k^2(G)$.*

PROOF. Clearly, $p(\boldsymbol{w}; G)$ is a multivariate normal. Denote by A the inverse covariance matrix of $p(\boldsymbol{w}; G)$. Then, expanding Eq. 3, we can show:

$$p(\boldsymbol{w}; G_\gamma) \propto \exp\left\{-\boldsymbol{w}' A_\gamma \boldsymbol{w}\right\}$$
$$A_\gamma = A + \gamma \Lambda_{ij}$$
$$[\Lambda_{ij}]_{pq} = \begin{cases} 1 & \text{if } p = q = i \text{ or } p = q = j \\ -1 & \text{if } p = i, q = j \text{ or } p = j, q = i \\ 0 & \text{otherwise} \end{cases}$$

Let M_i and M_j be the i^{th} and j^{th} columns of A^{-1}. Then, the covariance matrix A_γ^{-1} for G_γ depends on γ as follows:

$$\frac{\partial A_\gamma^{-1}}{\partial \gamma} = -(M_i - M_j)(M_i - M_j)'$$
$$\Rightarrow \frac{\partial \sigma_k^2(G_\gamma)}{\partial \gamma} = \left[\frac{\partial A_\gamma^{-1}}{\partial \gamma}\right]_{kk} = -(M_{ik} - M_{jk})^2 \leq 0$$

Thus, the variance of any $k \in V$ is a non-increasing function of γ. From the smoothness of matrix inversion, the result follows. □

This implies that addition of extra factors reduces node variances, so:

COROLLARY 1. *For any $G = (V, E)$ and $G' = (V', E')$ such that $V = V'$ and $E' \subseteq E$, $\sigma_k^2(G) \leq \sigma_k^2(G')$.*

In particular, consider the subgraph G' of G that consists of only the edges incident on i. This is a "star" graph, which is singly-connected and hence the sum-product message-passing algorithm is guaranteed to converge and be correct.

On this star graph G', let m_{ji} be the precision of the message from j to i at convergence[2]. Let τ_i represent the precision of node i: $\tau_i = 1/\sigma_i^2$. Then, we can show the following:

LEMMA 2. *For G', we have:*

$$\tau_i = \frac{1}{\phi_i^2} + \sum_{j \sim i} m_{ji}$$
$$m_{ji} = \frac{1}{2\sigma_{ij}^2}\left((\sigma_{ij}^2\tau_i + 2) - \sqrt{(\sigma_{ij}^2\tau_i)^2 + 4\frac{\tau_i}{\tau_j}}\right)$$

[2]More formally, if f represents the factor node connecting j and i, then there are messages from $j \to f$ and $f \to i$; we refer to the precision of $f \to i$ as m_{ji}.

PROOF SKETCH. Standard message-passing formulas yield:

$$m_{ji} = \frac{1}{\frac{1}{\tau_j - m_{ij}} + \sigma_{ij}^2} = \frac{1}{\tau_j - \frac{1}{\frac{1}{\tau_i - m_{ji}} + \sigma_{ij}^2}} \tag{4}$$

The result follows from the solution of this equation. □

We now have the tools to understand the problem inherent in the SocialPrior model. Consider a basic scenario where all social prior links are equally strong $\sigma_{ij}^2 = \sigma_{sp}^2$, and we want every node to have the same marginal $\tau_i = \tau$ before any data becomes available. Noting that precisions in G are greater than those in the subgraph G' (Lemma 1) yields the following basic corollaries.

COROLLARY 2. *If all nodes have the same precision $\tau_i = \tau > 0$, and node i has degree $\deg(i) \geq 2$, then:*

$$\sigma_{ij}^2\tau > \frac{\deg(i)\,(\deg(i) - 2)}{\deg(i) - 1}.$$

PROOF. Setting $y = \sigma_{ij}^2\tau$ and using Lemma 2:

$$\frac{2}{\deg(i)}y \geq (y + 2) - \sqrt{y^2 + 4} \quad \text{(setting } y = \sigma_{ij}^2\tau\text{)}$$

This is a quadratic equation that is satisfied only when

$$y \geq \frac{\deg(i)\,(\deg(i) - 2)}{\deg(i) - 1}.$$

□

COROLLARY 3. *If all nodes have the same precision $\tau_i = \tau > 0$, all social prior variances are identical $\sigma_{ij}^2 = \sigma_{sp}^2$, then*

$$\sigma_{sp}^2\tau > \frac{\Delta\,(\Delta - 2)}{\Delta - 1},$$

where Δ is the maximum degree of the graph and $\Delta \geq 2$.

Note that, by Lemma 1, these results hold for G even though they were proved using the subgraph G'.

The implication is that for a large social graph, the product of the social prior variance and the marginal precisions must be large ($\Delta \approx 5000$ for Facebook). A model can specify either a small σ_{sp}^2 or a small τ, but not both. A large σ_{sp}^2 means that inter-node ties are weak, and nullifies the reasons for using a social prior, while a small precision τ makes the model inflexible even before any data has been observed. Clearly, both situations are undesirable.

Even if the individual precisions τ_i are different, a similar result holds if all priors are the same $\phi_i = \phi > 0$.

COROLLARY 4. $\tau_i \geq \dfrac{1}{\phi^2} + \dfrac{\deg(i)}{\sigma_{sp}^2 + \phi^2}$.

PROOF SKETCH. Use $\tau_j \geq 1/\phi^2$ in Eq. 4. □

Again, either σ_{sp}^2 must be large, or τ_i.

Another way to see the problem is in terms of the *influence* of new data on the marginals. As above, consider the case of $\tau_i = \tau$ and $\sigma_{ij}^2 = \sigma_{sp}^2$. Suppose node i receives a message with precision $\delta\tau$ due to new observations.

LEMMA 3. *The change in precision of the neighbors of i in G' (with $\Delta \geq 2$) due to a $\delta\tau$ precision message to i is given by:*

$$\textit{influence} \leq \frac{\delta\tau}{(\Delta - 1)^2}.$$

PROOF SKETCH. When $\tau_i = \tau$, Lemma 2 yields $m_{ji} = m$ for all i, j and some constant m, and $\tau \geq \Delta m$. Now, the influence of the incoming message on some neighbor j of i is given by:

$$\text{influence} = \frac{1}{\frac{1}{\tau - m + \delta\tau} + \sigma_{sp}^2} - \frac{1}{\frac{1}{\tau - m} + \sigma_{sp}^2}$$

The result follows from analysis of this expression. □

Thus, the effect of new data at any node propagates minimally through the graph, irrespective of the model's exact specifications. Qualitatively similar results hold even for distinct σ_{ij}^2 and τ_i, as long as the sum $\tau_i = 1/\phi_i^2 + \sum_{j\sim i} m_{ji}$ is not dominated by any $O(1)$ subset of the neighbors of i. Note that this result is not restricted to one particular form of the prior; while Eq. 3 corresponds to an L_2 regularization, any norm will face similar problems due to the equivalence of norms.

INTUITION. The social prior is essentially a set of constraints:

$$w_i = w_{j1} + \mathcal{N}(0, \sigma_{sp}^2) = \ldots = w_{jn} + \mathcal{N}(0, \sigma_{sp}^2)$$
$$\Rightarrow w_{jk} = w_i + \mathcal{N}(0, \sigma_{sp}^2) \quad k = 1 \ldots n,$$

where $w_{jk} \sim w_i$. If the neighboring weights $w_{j1} \ldots w_{jn}$ were known, then w_i could simply be estimated as the average of these; by the Central Limit Theorem, the variance in this estimate would asymptotically be σ_{sp}^2/n. Since the variance is just $1/\tau_i$, we see that $\sigma_{sp}^2\tau_i \approx n = \deg(i)$ asymptotically, and Corollary 3 is exactly the finite-size form of this result. The generality of the Central Limit Theorem reinforces our previous remark regarding the result being unaffected by the exact model specifications.

4.2 The ShadowPrior Model

The above discussion highlighted two problems with the SocialPrior model: limited flexibility in picking σ_{sp}^2 and τ, and limited propagation of influence. We now propose a model which alleviates the first problem, and also provides a useful parametrization for social priors.

Ideally, we would like to keep σ_{sp}^2 small, so that social links have a strong effect on the joint distribution. However, as we have seen, this leads to high precision τ, which leads to extremely low data likelihood if the means of the w_i distributions are even slightly incorrect. To fix this, we introduce a new "shadow" variable w_i' for each w_i; the social prior is placed on the shadows w_i' while the "true" weights w_i are drawn from their shadows with an additional variance:

$$p(\boldsymbol{w} \mid \boldsymbol{w}') \propto \mathcal{N}\left(\text{diag}\left(\mathbf{0}\right), \text{diag}\left(\boldsymbol{\phi}^2\right)\right) \cdot \Pi_u \mathcal{N}\left(w_u - w_u'; 0, \psi^2\right)$$
$$p(\boldsymbol{w}') \propto \Pi_{u\sim v} \mathcal{N}\left(w_u' - w_v'; 0, \sigma_{sp}^2\right) \quad (5)$$

The model can be generalized to set different values of σ_{sp}^2 and ψ in the different terms; we call this model ShadowPrior.

It might seem as if ShadowPrior is merely a reparametrization of SocialPrior, and that if we marginalized out the shadow variables, we would get an instantiation of SocialPrior with specific σ_{ij}^2 values. However, this is not so.

LEMMA 4. *ShadowPrior is strictly more general than SocialPrior.*

PROOF. Every SocialPrior model can be trivially encoded as a ShadowPrior by setting all $\psi_i = 0$. To see that ShadowPrior is a strict superset, consider a node i and two of its neighbors j and k. By setting $\psi_i \to \infty$ in ShadowPrior, we can have $w_j, w_k \perp w_i$. In particular, if $\psi_j = \psi_k = \sigma_{i'j'}^2 = \sigma_{i'k'}^2 = 0$, then we get $w_j = w_k$ for any value of w_i. This is impossible in SocialPrior; to have $w_j = w_k$, we must set $\sigma_{ij}^2 = \sigma_{ik}^2 = 0$ which means that w_i must equal w_j. □

In ShadowPrior, only the shadow variables are involved in the social prior and hence can have high precisions τ_i', but the precisions τ_i of the "true" weights w_i can be different:

$$\tau_i = \frac{\tau_i'}{\alpha\left(\tau_i'\psi^2 + \alpha\right)} \qquad \alpha = \frac{\phi^2}{\phi^2 + \psi^2}$$

This is an increasing function of τ_i', but with an upper bound of $\tau_i = 1/\phi^2 + 1/\psi^2$ for very large τ_i'. Thus, the precision of the true weights is always under control, and the data likelihood does not fall too low.

However, this still does not solve the problem of limited influence propagation.

LEMMA 5. *In ShadowPrior, the change in precision of a neighbor j of i in G' (with $\Delta \geq 2$) due to a $\delta\tau$ precision message to i (with $\deg(i) \geq 2$) is:*

$$\textit{influence} \leq \delta\tau \cdot (\alpha) \left(\frac{1}{(\deg(i') - 1)^2}\left(\frac{\phi^2}{\sigma_{sp}^2} + 1\right)^2\right)\left(\frac{1}{\psi^2\tau_j + \alpha}\right)$$

where $\alpha = \dfrac{\phi^2}{\phi^2 + \psi^2}$.

Thus, the shadow variables enable us to have strong social links (low σ_{sp}^2) and still have relatively wide marginal precisions τ_i, which was impossible under SocialPrior. However, this is achieved only by breaking the direct connection between the weights w_i; now, the weights corresponding to $i \sim j$ are connected via a long path $w_i \sim w_i' \sim w_j' \sim w_j$, so the problem of poor influence propagation remains. The problem is that we have not fixed the underlying cause — the constraints placed by the Central Limit Theorem. For this, we must try a very different approach, discussed next.

4.3 The MixturePrior model

All models discussed above suffer from the problem of poor influence propagation: a increase in precision of $\delta\tau$ causes only $O(1/\deg^2)$ change in the neighboring precisions. This would cease to be a problem if we could only restrict the degrees of the nodes, e.g., by deleting all but the top-k most "important" social links for each node. Recalling the intuition from the analysis of SocialPrior, if only a constant k links are active, then the Central Limit Theorem does not apply, and hence the ill-effects on precisions do not materialize. The top-k neighbors may be picked manually, but they might not be optimal for the learning algorithm. Ideally, the algorithm should learn the top-k edges as well; this motivates our next model.

Instead of the normal social prior $\mathcal{N}\left(w_u - w_v; 0, \sigma_{sp}^2\right)$ (Eq. 2), the MixturePrior model uses a Gaussian mixture to represent social links:

$$z_{uv} \sim \text{Bernoulli}(\pi_{uv}) \quad (6)$$
$$w_u - w_v \mid z_{uv} = 1 \sim \mathcal{N}(0, L^2) \quad (7)$$
$$w_u - w_v \mid z_{uv} = 0 \sim \mathcal{N}(0, \sigma_{sp}^2) \quad (8)$$

where π_{uv} is the mixing proportion, and L is some large number that we will later send to infinity. Intuitively, the

link is "present" only when the hidden variable $z_{uv} = 0$, and the probability of this occurrence can be tuned via π_{uv}. For example, to retain at most k edges for every node, set:

$$\pi_{uv} = 1.0 - \min\left(k/\max(\deg(u), \deg(v)), 1.0\right). \quad (9)$$

COMPUTING THE MESSAGES. While MixturePrior expands the expressiveness of the basic model, the marginal distributions and the update messages are no longer Gaussian, but instead are Gaussian mixtures; a node with degree d will have a marginal distribution composed of 2^d Gaussians. Clearly, approximations are required, and we choose to approximate the marginals (and hence the messages across the social links) as Gaussians. However, instead of approximating each message independently, we use Expectation Propagation to jointly approximate the messages [12]. This has the property that for any single social link, the approximate messages for that link are the best approximation to the actual Gaussian mixture for that link, given the approximate messages for all other social links.

Paraphrasing Eqs. 6-8, the social link between each pair of neighbors i and j is represented by a factor:

$$\kappa(w_i, w_j) = \pi_{ij}\mathcal{N}(w_i - w_j; 0, L^2) + (1-\pi_{ij})\mathcal{N}(w_i - w_j; 0, \sigma_{sp}^2).$$

We approximate this factor as $\kappa(w_i, w_j) \approx t_{ij}^{(i)}(w_i) t_{ij}^{(j)}(w_j)$, where the two terms on the right are Gaussians that depend only on w_i or w_j respectively. The joint distribution of Eq. 3 is then approximated by

$$p(\boldsymbol{w}) = \mathcal{N}\left(\text{diag}\left(\boldsymbol{\mu}\right), \text{diag}\left(\boldsymbol{\phi}^2\right)\right) \Pi_{ij} \kappa(w_i, w_j) \quad (10)$$

$$\approx \mathcal{N}\left(\text{diag}\left(\boldsymbol{\mu}\right), \text{diag}\left(\boldsymbol{\phi}^2\right)\right) \Pi_{ij} t_{ij}^{(i)}(w_i) t_{ij}^{(j)}(w_j) \quad (11)$$

Intuitively, the approximate factors $t_{ij}^{(i)}(w_i)$ and $t_{ij}^{(j)}(w_j)$ correspond to Gaussian messages sent from j to i and vice versa (i.e., m_{ji} and m_{ij} respectively). In Expectation Propagation, we set the their parameters iteratively; given the current approximations for all other factors, $t_{ij}^{(i)}(w_i)$ and $t_{ij}^{(j)}(w_j)$ should be set so as to minimize the KL-divergence between (a) the distribution obtained by replacing the product $t_{ij}^{(i)}(w_i)\, t_{ij}^{(j)}(w_j)$ in Eq. 11 by the actual factor $\kappa(w_i, w_j)$, and (b) the approximate joint of Eq. 11.

For ease of notation, we shall use $\mathcal{N}^{-1}(.)$ to denote the normal distribution in terms of its "precision" and "precision-mean"; $\mathcal{N}^{-1}(x, y) = \mathcal{N}(x/y, 1/y)$. Then, let $N^{-1}(w_i; \alpha_i, \beta_i)$ be the product of all terms in Eq. 11 that depend on w_i , except $t_{ij}^{(i)}(w_i)$; these consist of the prior term $\mathcal{N}(\mu_i, \phi_i^2)$ and the approximate messages from all other neighbors of i. Let $N^{-1}(w_j; \alpha_j, \beta_j)$ be the corresponding product for w_j. Note that the products are Gaussian due to the assumed normality of the approximate messages. Then, as we take $L \to \infty$ (i.e., remove the link between i and j with probability π_{ij}), Expectation Propagation leads to the following equations:

$$m_{ji} = t_{ij}^{(i)}(w_i) = \mathcal{N}^{-1}(pm, p), \text{ where} \quad (12)$$

$$\frac{1}{\beta_i + p} = \frac{\pi_{ij}}{\beta_i} + \frac{1 - \pi_{ij}}{\beta_i + \gamma_j} \quad (13)$$

$$\frac{\alpha_i + pm}{\beta_i + p} = \pi_{ij}\frac{\alpha_i}{\beta_i} + (1 - \pi_{ij})\frac{\alpha_i + \gamma_j \frac{\alpha_j}{\beta_j}}{\beta_i + \gamma_j} \quad (14)$$

$$\gamma_j = \frac{1}{\frac{1}{\beta_j} + \sigma_{sp}^2} \quad (15)$$

The precision and precision-mean of $t_{ij}^{(j)}(w_j)$ can be computed similarly.

As a sanity check, consider the case of $\pi_{ij} = 0$, which corresponds to having all social links be connected in the social prior. In this case, we find:

$$p = \frac{1}{1/\beta_j + \sigma_{sp}^2} \quad pm = \frac{\alpha_j/\beta_j}{1/\beta_j + \sigma_{sp}^2} \quad (16)$$

which is exactly the message that would be computed for a social link under SocialPrior. Conversely, when $\pi_{ij} = 1$ (i.e., no social link), we find $p = pm = 0$, just as expected.

We can also compute the message as a function of the marginal precisions τ_i.

LEMMA 6. *If $\tau_i = \tau\ \forall i$ and $\pi_{ij} > 0$,*

$$p = \frac{y + 2 - \sqrt{y^2 + 4 + 4y\pi_{ij}}}{2\sigma_{sp}^2} \quad (y = \sigma_{sp}^2 \tau).$$

PROOF SKETCH. Proved using Eqs. 12-15. □

Note that, for consistency, the marginal precision must be at least the sum of the incoming messages:

$$\tau \geq \sum_{j \sim i} m_{ji} = \deg(i) p$$

$$\Rightarrow 2y \geq y + 2 - \sqrt{y^2 + 4 + 4y\pi_{ij}} \qquad (y = \sigma_{sp}^2 \tau)$$

which can be achieved for any degree $\deg(i)$ by setting π_{ij} close enough to 1. This allows flexibility in setting σ_{sp}^2 and τ, which was one of the weaknesses of SocialPrior.

As for the earlier models, we can compute the influence propagated from one node to another.

LEMMA 7. *In MixturePrior, with the mixing proportions π_{ij} set as in Eq. 9, the change in precision of a neighbor j of i in G' due to a $\delta\tau$ precision message to i ($\deg(i) \geq 2$) is:*

$$\textit{influence} = \delta\tau \cdot \frac{1 - \pi_{ij}}{\left(1 + \pi_{ij} + \beta\sigma_{sp}^2\right)^2} = \delta\tau \cdot O\left(\frac{1}{\deg(i)}\right)$$

PROOF. The proof follows easily from Eq. 13. □

Note that this $O\left(1/\deg(i)\right)$ rate compares favorably with the $O\left(1/\deg^2(i)\right)$ rate obtained from previous models.

Thus, we have finally achieved our desiderata: we can combine strong social priors (low σ_{sp}^2) with relatively wide marginals (small τ_i), and still have much better influence propagation than SocialPrior or ShadowPrior. This is because, intuitively, only a constant k of the links are active for any node in the social graph, so the corresponding k messages dominate the rest. Hence, the Central Limit Theorem no longer applies, freeing us from its strict constraint.

4.4 Disengaging the social prior

There is one final issue that affects all versions of the social prior discussed above. Typically, the effect of a prior diminishes over time, and eventually vanishes as enough data becomes available. However, this process is much slower with the social prior: stronger evidence from data is countered by stronger messages from neighbors, since the neighbors get new data as well. Thus, for any one feature, the social prior and the evidence from data remain roughly balanced throughout.

Since the goal of the social prior was to speed up learning, it should be "disengaged" once enough data is available to determine the feature weights from evidence alone. Hence, we

stop performing message updates to weights whose variance drops below a threshold, allowing it to change henceforth only through evidence from data.

4.5 Implementation Details

For our experiments, we implemented all the social prior models as part of an online learning system: for each new example, the algorithm first predicts the outcome (click or not), and then uses the true outcome to train. Model parameters must ensure that the initial precisions τ are small enough, or else learning from data would be too slow; we set $\tau = 1$. By Corollary 2, this lower-bounds σ_{ij}^2 for both SocialPrior and ShadowPrior; in fact, if we want $\sigma_{ij}^2 = \sigma_{sp}^2$, then $\sigma_{sp}^2 \approx \Delta/\tau = 5000$ for Facebook (Corollary 3). Only for a line graph ($\Delta = 2$) places no constraints on σ_{sp}^2.

On the other hand, this problem never arises for MixturePrior, as we saw from the discussion after Lemma 6. Here, we can use any combination of τ and σ_{sp}^2, as long as π_{ij} is set as in Eq. 9; we use $k = 3$, meaning only 3 top social links will be (probabilistically) "active".

5. BROADER APPLICATIONS

While the discussion until now has focused on the social-features context, the methods we derived are far more widely applicable. Indeed, what we have is a framework for learning feature weights with known pairwise relationships. We shall look at two such cases, with features derived from (a) histograms bins, and (b) trees.

5.1 Binned features

Continuous-valued features, such as a user's age or average time spent on a web portal, can be incorporated in a learning framework by converting them into categorical features by binning them, and then learning a separate weight for each bin. If the number of bins is large enough, this can closely approximate the true effect of the feature. However, the weight for each bin must be learned independently. This forces a trade-off between time to convergence (better with fewer bins) against prediction accuracy (better with more bins). In fact, this problem exists for any ordinal feature that is used as a categorical variable.

The root cause of the problem is that the ordering of the bins is forgotten, and the smoothness of feature weights as a function of feature values is ignored. However, our methods can be easily applied here: each bin is linked to its adjacent bins (its "friends") by the social prior. This creates a line graph, with correspondingly simple message-passing updates. In fact, message-passing is guaranteed to converge correctly on this graph.

5.2 Trees

Histogram bins are simply intervals in feature space on which pairwise distances can be specified. Intervals can be easily generalized to "regions" of feature space, as long as some reasonable distance between regions exists.

For example, if online content can be placed into a known taxonomy, then the leaves of the taxonomy represent such regions, with the tree metric giving a distance between them. Thus, each leaf of the taxonomy becomes a feature, and a "social" prior can connect close leaves, with the prior variance σ_{sp}^2 depending on the distance metric.

A similar idea holds for, say, leaves of a random forest or boosted decision trees created using some historical data. These leaves can then be used as extra features in a learning algorithm, possibly in addition to the features on which the trees were originally built. However, in this case, leaves from the same tree should not be connected, since the tree construction algorithm specifically tries to find regions in feature space that are maximally distinct. Thus, the tree metric is inapplicable, and we need some metric between leaves of different trees.

Noting that each leaf in a decision tree corresponds to a set of (possibly open-ended) feature intervals, we propose using the fraction of overlap between these intervals as a measure of similarity between leaves. Only leaves that reference the same set of features are comparable in this manner; however, evidence from a large real-world content-serving system shows that even with this restriction, many leaf pairs show significant overlap. The "social graph" between leaves is no longer a simple line graph, as with the histogram bins above, but is rather close to a set of quasi-cliques. Another related distance measure is the closest L_1 distance between any two points from the two leaves, with the distance being 0 for overlapping leaves. In our experiments, we find that both these distance measures yield similar results.

6. EXPERIMENTS

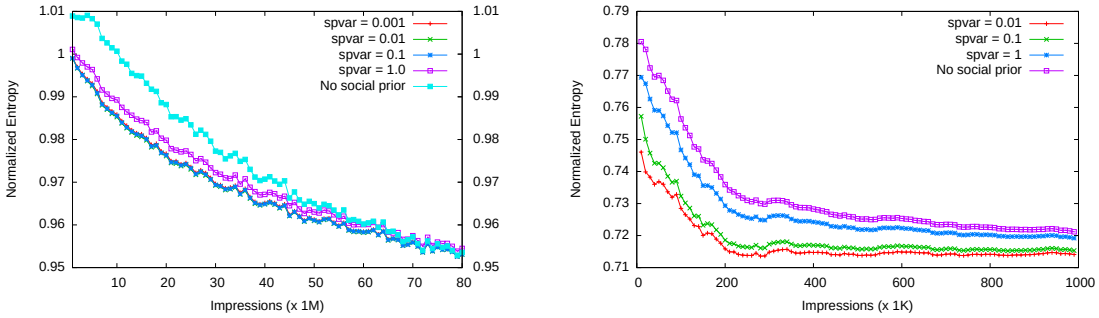
The primary purpose of the social prior was to speed up the learning process, and this is the question we investigate next. We present experiments on two problem settings. The first, called Feeds, involves predicting clicks on Facebook news-feed stories, with the user-ID as a feature. The data consists of a sample of around $100M$ stories presented to a $3.5M$-strong subset of Facebook users, with around $410M$ undirected friendship edges between them. The second, called Ads, involves predicting clicks on ads using 18 different ordinal features bucketed into 100 bins each. In both problems, training and prediction are performed online: for each new example, the algorithm first predicts the outcome (click or not), and then uses the true outcome to train. Clicks, non-clicks, and the set of users are all sampled separately, so the reported accuracy should not be thought of as the "true" accuracy of the current Facebook systems.

We measure accuracy with Normalized Entropy (NE), defined as the ratio of model log-likelihood to the observed entropy of outcomes (click or not). A NE of 1.0 corresponds to a random predictor that knows the correct fraction of clicks in the dataset, and lower values are better.

Accuracy on Feeds.

For Feeds, the weights for the user-ID features are related via the social network which has a maximum degree of $\Delta = 5000$. As discussed in Section 4.5, we must have $\sigma_{sp}^2 > \Delta/\tau$ for both SocialPrior and ShadowPrior, and even a modest initial marginal precision of $\tau = 1.0$ implies $\sigma_{sp}^2 > 5000$, and translates into almost no influence of a weight on its neighbors (Lemmas 3 and 5). Indeed, in our experiments on Feeds, the results from SocialPrior and ShadowPrior were indistinguishable from NoSocial, even after using the smallest σ_{ij}^2 allowable by Corollary 2; hence, we do not separately show their results here. Only MixturePrior can reasonably decouple τ and σ_{sp}^2, and is the only applicable model.

Figure 1(a) shows the results for various settings of σ_{sp}^2 for Feeds. We observe that any social prior at all is better than NoSocial, with $\sigma_{sp}^2 = 0.01$ being the best ($\sigma_{sp}^2 = 0.001$ is also almost identical). The benefits of a social prior are greatest when the fewest training examples have been seen, which is



(a) MixturePrior on Feeds (b) SocialPrior on Ads

Figure 1: *Normalized entropy*: Accuracy for various σ_{sp}^2

when there is the least evidence from data. The gain can be characterized by the number of extra training examples needed by NoSocial to replicate the accuracy of MixturePrior: in Feeds, this is around $12M$ training examples. Indeed, MixturePrior remains better than NoSocial even after $60M$ examples. Both of these demonstrate the impressive utility of the social prior.

Also, below a certain σ_{sp}^2 (≈ 0.01), there are very little differences. Thus, choosing the "optimal" σ_{sp}^2 is not necessary as long as it is in the right range.

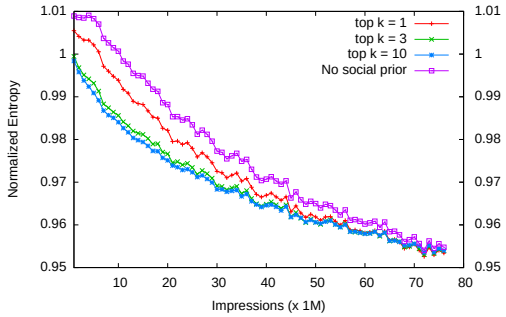


Figure 2: *Effect of the top k in* MixturePrior

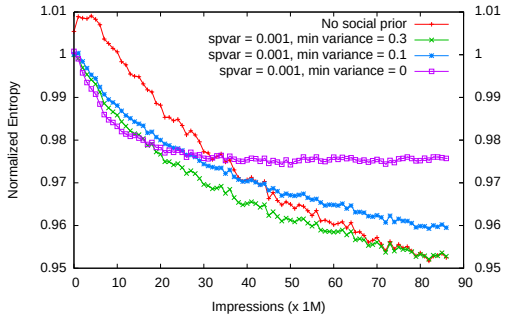


Figure 3: *NE against the variance threshold for* Feeds*:* Without a threshold, the social prior is too restrictive when enough data becomes available.

ACCURACY ON ADS. Recall that the Ads problem has binned features, whose relationship structure is given by a line graph (each bin is similar to its adjacent bins). Since the maximum degree $\Delta = 2$, the constraint of Corollary 3 is always satisfied, so SocialPrior can use any σ_{sp}^2. Hence, in this case, we report results using SocialPrior; the more complicated models can offer no further gains.

Figure 1(b) shows the results. Once again, the social prior leads to significantly faster learning, with $\sigma_{sp}^2 = 0.001$ being optimal but $\sigma_{sp}^2 = 0.01$ being almost as good, implying that picking the optimal value is not as important as being in the correct range. Note that NoSocial does not reach the accuracy for SocialPrior with $\sigma_{sp}^2 = 0.01$ even after $1M$ training examples, and its rate of improvement is very slow. Thus, we can conclude that the social prior yields significant speed-up not only in the initial stages of learning, but also in reaching convergence.

SENSITIVITY TO THE TOP k. Figure 2 plots NE as a function of the top k important edges used in the mixing proportion π_{ij} in MixturePrior (Eq. 9). The experiments are for Feeds, with $\sigma_{sp}^2 = 0.01$ everywhere. We see that $k = 3$ is nearly optimal, and there is not much gained by increasing k. This suggests that at least 3 "close" friends are needed to infer a user's interests, and all our experiments use $k = 3$.

IMPORTANCE OF DISENGAGEMENT. Figure 3 shows the importance of the variance threshold below which the social prior is "disengaged." Without any threshold, the NE decreases the fastest initially, but stops improving far too early; this is because the social prior forces neighboring weights to move in lockstep, even if data indicates otherwise. Conversely, a high threshold makes the model equivalent to NoSocial, which performs poorly initially. In our case, a variance threshold of 0.3 performs best.

EFFECT OF FARTHER PROPAGATION. Instead of propagating messages from a node receiving new training data to its immediate neighbors only, we could also propagating further. However, this could slow down training, and hence the prediction throughput, in our online prediction setting. Hence, we consider two alternatives where propagation is done in a separate thread: (a) either all propagation of messages could be done in a separate thread, or (b) messages to immediate neighbors could be spread immediately, with further propagation being performed in a thread. Figure 4 shows the results for Ads. We see that (a) any form of propagation greatly outperforms the NoSocial case, and (b) propagating to just the immediate neighbors is almost as good as further propagations, while requiring the least CPU. This justifies the approach taken in the previous experiments.

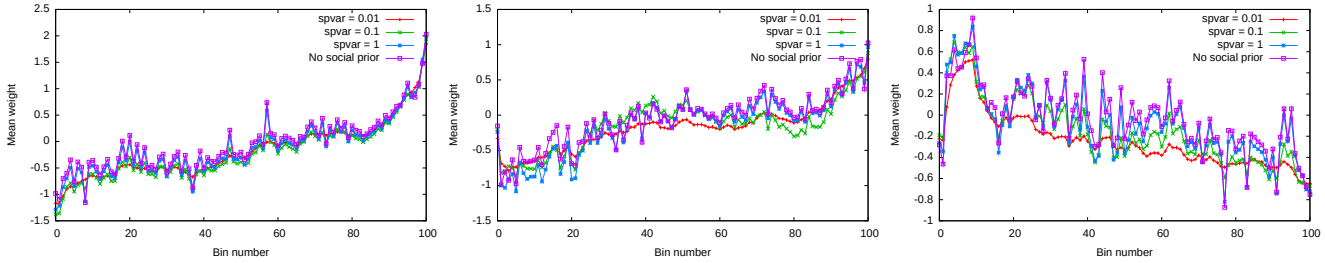


Figure 5: ***Mean weight for each bin in* Ads:** **Smaller the social prior variance, smoother the plot.**

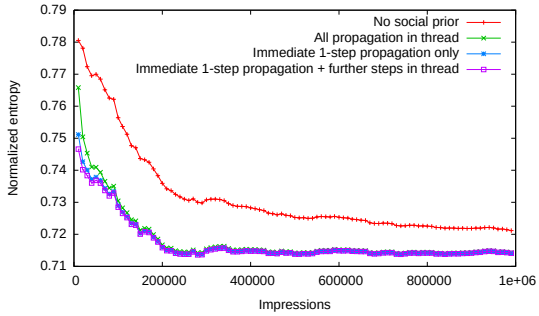


Figure 4: ***Propagating messages to immediate neighbors is almost as beneficial as propagating further.***

SMOOTHING OF WEIGHTS. The social prior pulls related features closer together. To verify this, we plot the mean weight as a function of the bin number for several binned features in Ads (Figure 5). Since the binned features in Ads form a line graph, we expect a strong social prior (i.e., smaller values of σ_{sp}^2) to lead to weights that vary smoothly as a function of bin number, and this is indeed observed.

7. CONCLUSIONS

Large-scale learning can be speeded up considerably if we can utilize known relationships between features; if two features are similar, then any new data affecting one can be propagated to improve our knowledge of the other. We proposed a "social prior" to achieve this in a Bayesian setting. Via analysis of this model, we showed that obvious instantiations of such a prior suffer from a significant weakness: any reasonably strong link between features forces the marginal variances to become too small, even in the absence of data. We showed how a two-component mixture model can solve this problem, and presented update equations based on Expectation Propagation for learning under this model. Experiments on large real-world click prediction problems on a subset of the Facebook social network shows that our method can save $12M$ training examples in the initial part of the learning. On a separate problem, a social prior on binned features converges in less than a fifth of the time required by the baseline model. This clearly demonstrates the utility of the social prior.

8. REFERENCES

[1] W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. In *STOC*, 2000.

[2] E. Bakshy, D. Eckles, R. Yan, and I. Rosenn. Effects of social cues and tie strength in social advertising: Evidence from field experiments. In *EC*, 2012.

[3] H. Bao and E. Chang. Adheat: An influence-based diffusion model for propagating hints to match ads. In *WWW*, 2010.

[4] D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

[5] M. Gartrell, U. Paquet, and R. Herbrich. A bayesian treatment of social links in recommender systems. Technical report, Univ. of Colorado, 2012.

[6] S. Goel and D. Goldstein. Predicting individual behavior with social networks, 2012.

[7] 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*, 2010.

[8] J. M. Hernandez-Lobato, D. Hernandez-Lobato, and A. Suarez. Network-based sparse bayesian classification. *Pattern Recognition*, 44, 2011.

[9] I. Konstas, V. Stathopoulos, and J. Jose. On social networks and collaborative recommendation. In *SIGIR*, 2009.

[10] Y. Lu, P. Tsaparas, A. Ntoulas, and L. Polanyi. Exploiting social context for review quality prediction. In *WWW*, 2010.

[11] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King. Recommender systems with social regularization. In *WSDM*, 2011.

[12] T. Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, MIT, 2001.

[13] T. Sandler, P. P. Talukdar, L. H. Ungar, and J. Blitzer. Regularized learning with networks of features. In *NIPS*, 2008.

[14] C. Wang, R. Raina, D. Fong, D. Zhou, J. Han, and G. Badros. Learning relevance from heterogeneous social network and its application in online targeting. In *SIGIR*, 2011.

[15] S. Wang, L. Yuan, Y.-C. Lai, X. Shen, P. Wonka, and J. Ye. Feature grouping and selection over an undirected graph. In *KDD*, 2012.

[16] Y. Weiss and W. T. Freeman. Correctness of belief propagation in gaussian graphical models of arbitrary topology. *Neural Computation*, 13:2173–2200, 2001.

[17] Z. Wen and C.-Y. Lin. On the quality of inferring interests from social neighbors. In *KDD*, 2010.