A Probabilistic Model for Using Social Networks in Personalized Item Recommendation

Allison J.B. Chaney
Princeton University
achaney@cs.princeton.edu

David M. Blei
Columbia University
blei@cs.columbia.edu

Tina Eliassi-Rad
Rutgers University
eliassi@cs.rutgers.edu

ABSTRACT

Preference-based recommendation systems have transformed how we consume media. By analyzing usage data, these methods uncover our latent preferences for items (such as articles or movies) and form recommendations based on the behavior of others with similar tastes. But traditional preference-based recommendations do not account for the social aspect of consumption, where a trusted friend might point us to an interesting item that does not match our typical preferences. In this work, we aim to bridge the gap between preference- and social-based recommendations. We develop *social Poisson factorization* (SPF), a probabilistic model that incorporates social network information into a traditional factorization method; SPF introduces the social aspect to algorithmic recommendation. We develop a scalable algorithm for analyzing data with SPF, and demonstrate that it outperforms competing methods on six real-world datasets; data sources include a social reader and Etsy.

Keywords

Recommender systems; probabilistic models; social networks.

1. INTRODUCTION

Recommendation has become a core component in our online experience, such as when we watch movies, read articles, listen to music, and shop. Given information about what a user has consumed (e.g., items viewed, marked as "favorites," or rated), the goal of recommendation is to suggest a set of unobserved items that she will like.

Most recommendation systems aim to make personalized suggestions to each user based on similar users' histories. To solve this problem, matrix factorization algorithms are the workhorse methods of choice [20, 32]. Factorization algorithms use historical data to uncover recurring patterns of consumption, and then describe each user in terms of their varying preferences for those patterns. For example, the discovered patterns might include art supplies, holiday decorations, and vintage kitchenware; and each user has different preferences for each category. To perform recommendation, factorization algorithms find unmarked items of each user that are characteristic of her preferences.

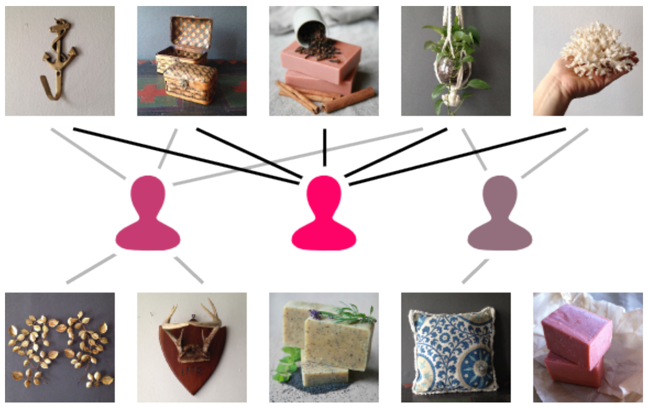

Figure 1: Observed and recommended items[1] for an Etsy user. The user is shown in the center, with friends on the sides. The top row is training items and the bottom row is the top recommendations from our model (SPF). Some items are recommended because they are favorites of the friends, and others because they match the general preferences of the user.

Many applications of recommendation contain an additional source of information: a social network. This network is increasingly available at the same platforms on which we read, watch, and shop. Examples include Etsy, Instagram, and various social readers. Researchers have found that users value the opinions of their friends for discovering and discussing content [18, 33], and online access to their network can reinforce this phenomenon.

Factorization approaches, however, cannot exploit this information. They can capture that you may enjoy an item because it matches your general preferences, but they cannot capture that you may enjoy another because your friend enjoyed it. Knowing your connections and what items your friends like should help better predict what you will enjoy.

In this paper we develop *social Poisson factorization* (SPF), a new Bayesian factorization method that accounts for the social aspect of how users consume items. (SPF is based on Poisson factorization [11], a new model that is particularly suited for implicit data.) SPF assumes that there are two signals driving each user's clicks: her latent preferences for items (and the latent attributes of each) and the latent "influence" of her friends.[2] From observed

[1]Etsy product images courtesy of Amber Dubois and Ami Lahoff. Used with permission.

[2]There is a large body of research literature on peer influence [22, 6, 30]. In this work we use the term to indicate the latent change in consumption due to social connections.

data—which contains both click histories and a social network—SPF infers each user's preferences and influences. Subsequently, it recommends items relating both to what a user is likely to be interested in and what her friends have clicked.

Figure 1 gives the intuition. The user is in the center. She clicked on items (on the top, connected to the user), has friends (to either side), and those friends have clicked on items too (top and bottom, connected to each friend). From this data, we can learn both about her preferences (e.g., for handmade soap) and about how much she is influenced by each of her friends (e.g., more strongly by the friend on the left). SPF recommends items on the bottom, based on both aspects of the data. It is important to be able to explain the origins of recommendations to users [15], and SPF can tell the user why an item was recommended: it can indicate friends ("you always trust Sally") and general item attributes ("you seem to like everything about ninjas") to describe the source of recommendations.

We use the language of users clicking on items. This is just a convenience—our model applies just as easily for users purchasing, rating, watching, reading, and "favoriting" items. Our goal is to predict which of the unclicked items a user will want to click.

In the following, we develop the mathematical details behind the model (Section 2), derive an efficient learning algorithm (based on variational inference) for estimating it from data (Section 2, Appendix), and evaluate it on six real-world data sets (Section 3). In all cases, our social recommendation outperforms both traditional factorization approaches [11, 29] and previous recommendation methods that account for the network [14, 17, 24, 25, 34].

Related work. We first review previous research on using social networks to help recommend items to users. A crucial component of SPF is that it infers the influence that users have with each other. In previous work, some systems assume that user influence (sometimes called "trust") is observed [27]. However, trust information beyond a binary yes/no is onerous for users to input, and thus observing trust beyond "following" or "friending" is impractical in a large system. Others assume that trust is propagated [2] or computed from the structure of the network [10]. This is limited in that it ignores user activity, which can reveal the trust of a user for some parts of the network over others; SPF captures this idea. Information diffusion [8, 12] also relies on user activity to describe influence, but focuses on understanding the widespread flow of information. A final alternative is to compute trust from rating similarities between users [9]. However, performing this computation in advance of fitting the model confounds general preference similarity with instances of influence—two people with the same preferences might read the same books in isolation.

Other research has included social information directly into various collaborative filtering methods. Ref. [36] incorporates the network into pairwise ranking methods. Their approach is interesting, but one-class ranking methods are not as interpretable as factorization, which is important in many applications of recommender systems [15]. Refs. [25, 28, 34] have explored how traditional factorization methods can exploit network connections. For example, many of these models factorize both user-item data and the user-user network. This brings the latent preferences of connected users closer to each other, reflecting that friends have similar tastes. Refs [24, 35] incorporate this idea more directly by including friends' latent representations in computing recommendations made for a user.

Our model has a fundamentally different approach to using the network to form recommendations. It seeks to find friends with different preferences to help recommend items to a user that are outside of her usual taste. For example, imagine that a user likes an item simply because many of her friends liked it too, but that it falls squarely outside of her usual preferences. Models that adjust their friends' overall preferences according to the social network do not allow the possibility that the user may still enjoy this anomalous item. As we show in Section 3, using the social network in this way performs better than these previous approaches.

2. SOCIAL POISSON FACTORIZATION

In this section we develop social Poisson factorization (SPF). SPF is a model for recommendation; it captures patterns in user activity using traditional signals—latent user preferences and latent item attributes—and estimates how much each user is influenced by his or her friends' observed clicks. From its estimate of influence, SPF recommends clicked items by influential friends even when they are not consistent with a user's factorization-based preferences.

We first review Poisson factorization and give the intuition on our model. Then, we formally specify our model, describe how to form recommendations, and discuss how we learn the hidden variables.

Background: Poisson factorization. SPF is based on Poisson factorization (PF) [11], a recent variant of probabilistic matrix factorization for recommendation. Let r_{ui} be the count of how many times user u clicked item i.[3] PF assumes that an observed count r_{ui} comes from a Poisson distribution. Its rate is a linear combination of a non-negative K-vector of user preferences θ_u and a non-negative K-vector of item attributes β_i,

$$r_{ui} \sim \text{Poisson}(\theta_u^\top \beta_i).$$

The user preferences and item attributes are hidden variables with Gamma priors. (Recall that the Gamma is an exponential family distribution of positive values.) Given a matrix of observed clicks, posterior inference of these hidden variables reveals a useful factorization: latent attributes describe each item and latent preference describe each user. These inferences enable personalized recommendations.

PF relates to the GaP topic model [5], and can be viewed as a type of Bayesian non-negative matrix factorization [21]. Ref. [11] shows that PF realistically captures patterns of user behavior, lends itself to scalable algorithms for sparse data, and outperforms traditional matrix factorization based on Gaussian likelihoods [11, 29].

Social Poisson factorization. In many settings, users are part of an online social network that is connected to the same platforms on which they engage with items. For some, such as Etsy, these networks are innate to the site. Others may have external data, e.g., from Facebook or LinkedIn, about the network of users.

We build on PF to develop a model of data where users click on items and where the same users are organized in a network. Social Poisson factorization (SPF) accounts for both the latent preferences of each user and the click patterns of her neighbors.

Consider the user whose items are shown in Figure 1. The intuition behind SPF is that there can be two reasons that a user might like an item. The first reason is that the user's general preferences match with the attributes of the item; this is the idea behind Poisson factorization (and other factorization approaches). For example, the user of Figure 1 may inherently enjoy handmade soap. A second reason is that the user has a friend who likes the item, or perhaps a collection of friends who all like it. This possibility is not exposed by factorization, but captures how the user might find items that are outside of her general preferences. Without learning the influence of friends in Figure 1, the system could easily interpret the woven box as a general preference and recommend more boxes, even if the user doesn't usually like them.

[3]The theory around PF works on count data, but Ref. [11] shows that it works well empirically with implicit recommendation data, i.e., censored counts, as well.

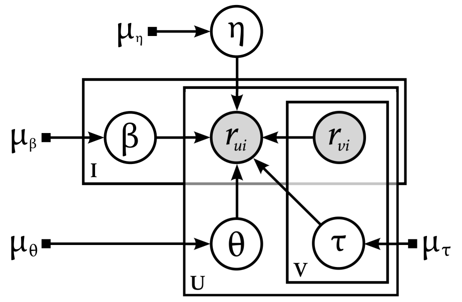

Figure 2: A conditional directed graphical model of social Poisson Factorization (SPF) to show considered dependencies. For brevity, we refer to the set of priors a and b as μ; for example, $\mu_\theta = (a_\theta, b_\theta)$. These hyperparameters are fixed.

SPF captures this intuition. As in PF, each user has a vector of latent preferences. However, each user also has a vector of "influence" values, one for each of her friends. Whether she likes an item depends on both signals: first, it depends on the affinity between her latent preferences and the item's latent attributes; second, it depends on whether her influential friends have clicked it.

Model specification. We formally describe SPF. The observed data are user behavior and a social network. The behavior data is a sparse matrix $\mathbf{R}$, where r_{ui} is the number of times user u clicked on item i. (Often this will be one or zero.) The social network is represented by its neighbor sets; $N(u)$ is the set of indices of other users connected to u. Finally, the hidden variables of SPF are per-user K-vectors of non-negative preferences θ_u, per-item K-vectors of non-negative attributes β_i, and per-neighbor non-negative user influences τ_{uv}. Loosely, τ_{uv} represents how much user u is influenced by the clicks of her neighbor, user v. (Note we must set the number of components K. Section 3 studies the effect of K on performance; usually we set it to 50 or 100.)

Conditional on the hidden variables and the social network, SPF is a model of clicks r_{ui}. Unlike many models in modern machine learning, we specify the joint distribution of the entire matrix $\mathbf{R}$ by the conditionals of each cell r_{ui} given the others,

$$r_{ui} \mid r_{-u,i} \sim \text{Poisson}\left(\theta_u^\top \beta_i + \textstyle\sum_{v \in N(u)} \tau_{uv} r_{vi}\right), \qquad (1)$$

where $r_{-u,i}$ denotes the vector of clicks of the other users of the ith item.[4] This equation captures the intuition behind the model, that the conditional distribution of whether user u clicks on item i is governed by two terms. The first term, as we said above, is the affinity between latent preferences θ_u and latent attributes β_i; the second term bumps the parameter up when trustworthy neighbors v (i.e., those with high values of τ_{uv}) also clicked on the item. Figure 2 shows the dependencies between the hidden and observed variables as a conditional graphical model.

To complete the specification of the variables, we place gamma priors on all of the hidden variables. We chose the hyperparameters of the gammas so that preferences, attributes, and influences are sparse. (See Section 3 for details.)

Forming recommendations with SPF. We have specified a probabilistic model of hidden variables and observed clicks. Given a $U \times I$ click matrix $\mathbf{R}$ and a $U \times U$ social network $\mathbf{N}$, we analyze the data by estimating the posterior distribution of the hidden preferences, attributes, and influences $p(\theta_{1:U}, \beta_{1:I}, \tau_{1:U} \mid \mathbf{R}, \mathbf{N})$. This posterior places high probability on configurations of preferences, attributes, and influence values that best describe the observed clicks within the social network.

From this posterior, we can form predictions for each user and each of their unclicked items. For a user u and an unclicked item j, we compute

$$\mathrm{E}\left[r_{uj}\right] = \mathrm{E}\left[\theta_u\right]^\top \mathrm{E}\left[\beta_j\right] + \sum_{v \in N(u)} \mathrm{E}\left[\tau_{uv}\right] r_{vj}, \qquad (2)$$

where all expectations are with respect to the posterior. For each user, we form recommendation lists by making predictions for the user's set of unclicked items and then ranking the items by these continuous-valued predictions. This is how we can use SPF to form a recommendation system.

Learning the hidden variables with variational methods. Social PF enjoys the benefits of Poisson factorization and accounts for the network of users. However, using SPF requires computing the posterior. Conditioned on click data and a social network, our goal is to compute the posterior user preferences, item attributes, and latent influence values.

As for many Bayesian models, the exact posterior for SPF is not tractable to compute; approximating the posterior is our central statistical and computational problem. We develop an efficient approximate inference algorithm for SPF based on variational methods [4, 19], a widely-used technique in statistical machine learning for fitting complex Bayesian models.[5] With our algorithm, we can approximate posterior expectations with very large click and network data (see Section 3).

Variational inference approximates the posterior by solving an optimization problem. We define a freely parameterized distribution over the hidden variables, and then fit its parameters to be close to the posterior distribution. We measure "closeness" by the Kullback-Leibler divergence, which is an assymetric measure of distance between distributions. Finally, we use the fitted variational distribution as a proxy for the posterior, for example to compute the expectations we need on the right-hand side of Eq. 2.

We use the *mean-field* variational family, where each latent variable is independent and governed by its own varitional parameter. The latent variables are the user preferences θ_u, item attributes β_i, and user influences τ_{uv}. The variational family is

$$q(\theta, \beta, \tau) = \prod_{u,k} q(\theta_{uk} | \lambda^\theta_{uk}) \prod_{i,k} q(\beta_{ik} | \lambda^\beta_{ik}) \prod_{u,v} q(\tau_{uv} | \lambda^\tau_{uv}). \qquad (3)$$

This is a flexible family. For example each cell of each user's preference vector θ_{uk} is associated with its own variational parameter λ^θ_{uk}. Thus, when fit to be close to the model's posterior, the variational parameters can capture each user's unique interests, each item's unique attributes, and each friend's unique influence value.

With the family in place, variational inference solves the following optimization problem,

$$q^*(\theta, \beta, \tau) = \arg\min_q \mathrm{KL}\left(q(\theta, \beta, \tau) || p(\theta, \beta, \tau \mid \mathbf{R}, \mathbf{N})\right). \qquad (4)$$

Note that the data—the clicks and the network—enter the variational distribution through this optimization. Finally, we use the resulting variational parameters of $q^*(\cdot)$ as a proxy for the exact posterior. This lets us use SPF to perform recommendation.

In the appendix we describe the details of how we solve the problem in Eq. 4 to find a local optimum of the KL divergence. We

[4]We are specifying an exponential family model conditionally. This leads to a well-defined joint if and only if the natural parameters for each conditional are sums and products of the sufficient statistics of the corresponding conditionals of the conditioning set [3]. In our case, this is satisfied.

[5]Source code available at https://github.com/ajbc/spf.

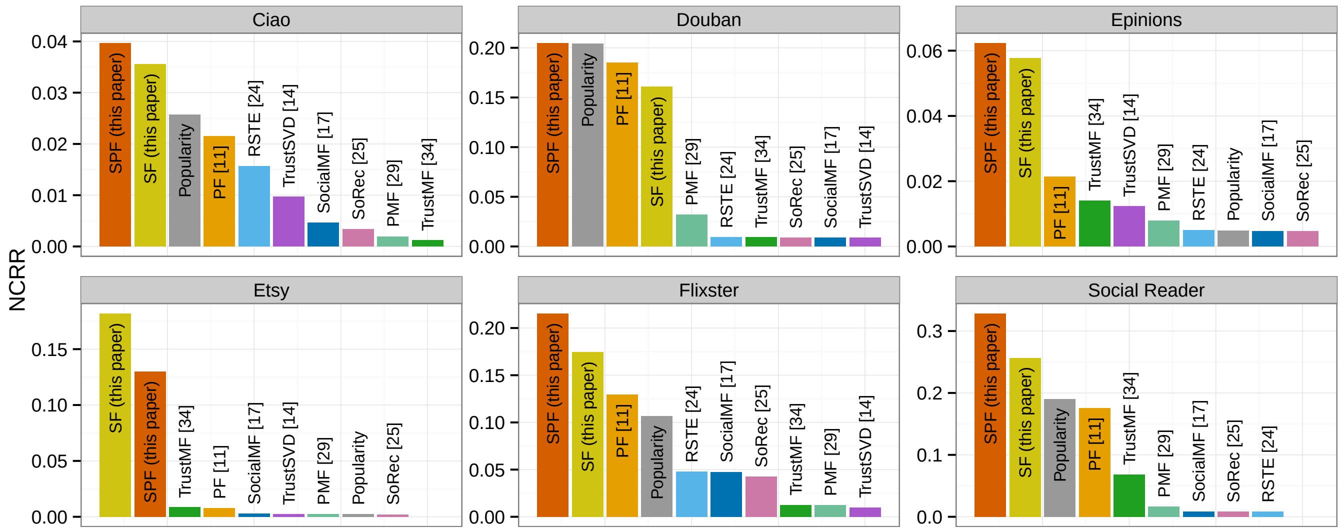

Figure 3: Performance of various methods on all six datasets, measured as NCRR averaged over users with held-out data. The Poisson-based factor models (PF and SPF) use $K = 40$ on Ciao, $K = 125$ on Epinions, $K = 100$ on Etsy, and $K = 50$ on Flixster, Douban, and Social Reader. Similar K values are used for competing models, but some perform best with lower K, in which case those settings are used. Models are sorted by performance. RSTE was omitted on Etsy data due to long run time and TrustSVD was omitted on Social Reader data due to difficulty in finding appropriate parameter settings. SPF outperforms all competing methods, except on Etsy, where our alternate model SF achieves top performance.

use a form of alternating minimization, iteratively minimizing the KL divergence with respect to each of the variational parameters while holding the others fixed. This leads to a scalable iterative algorithm, where each iteration runs on the order of the number of non-zero entries of the matrix. (In Section 3 we empirically compare the runtime of SPF with competing methods.) We now turn to an empirical study of SPF.

3. EMPIRICAL STUDY

In this section we study the performance of SPF. We compared SPF to five competing methods that involve a social network in recommendation [14, 17, 24, 25, 34] as well as two traditional factorization approaches [11, 29]. Across six real-world datasets, our methods outperformed all of the competing methods (Figure 3). We also demonstrate how to use SPF to explore the data, characterizing it in terms of latent factors and social influence. Finally, we assess sensitivity to the number of latent factors and discuss how to set hyperparameters on the prior distributions.

3.1 Datasets, methods, and metrics

Datasets and preprocessing. We studied six datasets. Table 1 summarizes their attributes. The datasets are:

- *Ciao* (ciao.co.uk) is a consumer review website with an underlying social network. Guo et al. [13] crawled DVD ratings and trust values for a small dataset of 7K users and 98K items.
- *Epinions* (epinions.com) is another consumer reviews website where users rate items and mark users as trustworthy. Our data source was Massa and Avesani [27]; the dataset consists of 39K users and 131K items.
- *Flixster* (flixster.com) is a social movie review website crawled by Jamali and Ester [17]. We binarized ratings, thresholding at 3 or above, resulting in 132K users and 42K items.
- *Douban* (douban.com) is a Chinese social service where users record ratings for music, movies, and books; it was crawled by Ma et al. [26]. It contains 129K users and 57K items.
- *Etsy* (etsy.com) is a marketplace for handmade and vintage items, as well as art and craft supplies. Users may follow each other and mark items as favorites. This data was provided directly by Etsy, and culled to users who have favorited at least 10 items and have at least 25% of their items in common with their friends; we omitted any items with fewer than 5 favorites. This is a large dataset of 40K users and 5.2M items.
- *Social Reader* is a dataset from a large media company that deployed a reader application on a popular online social network. The data contains a friendship network and a table of article clicks. We analyzed data from April 2-6, 2012, only including users who read at least 3 articles during that time. It contains 122K users and 6K items.

These datasets include both explicit ratings on a star scale and binary data. Content consumption is binary when the data is implicit (a news article was viewed) or when the system only provides a binary flag (favoriting). With implicit data, non-Poisson models require us to subsample 0's so as to differentiate between items; in these instances, we randomly sampled negative examples such that each user has the same number of positive and negative ratings. Note that Poisson-based models implicitly analyze the full matrix without needing to pay the computational cost of analyzing the zeros [11].

For each dataset, we preprocessed the network. We removed network connections where the users have no items in common. Note this advantages both SPF and comparison models (though SPF can learn the relative influence of the neighbors).

Our studies divided the data into three groups: approximately 10% of 1000 users' data are held-out for post-inference testing, 1% of all users' data are used to assess convergence of the inference algorithm (see Appendix), and the rest is used to train. One exception is Ciao, where we used 10% of all users' data to test.

	Ciao	**Epinions**	**Flixster**	**Douban**	**Social Reader**	**Etsy**
# of users	7,375	39,307	131,542	129,097	121,950	39,862
# of items	97,540	130,786	41,878	56,862	6,153	5,201,879
# user-item interactions	270,427	639,775	6,740,332	16,207,151	489,735	18,650,632
% user-item interaction matrix	0.038%	0.012%	0.122%	0.221%	0.065%	0.009%
interaction type	5-star	5-star	binary (thresholded)	5-star	binary (clicks)	binary (favoriting)
network type	directed	directed	undirected	undirected	undirected	directed
# network connections	56,267	176,337	488,869	1,323,828	100,175	4,761,437
network edge density	0.103%	0.011%	0.006%	0.016%	0.001%	0.300%
% shared	25.0%	36.0%	62.3%	51.0%	50.1%	30.8%

Table 1: Attributes of each data source, post-curation. User-item interactions are non-zero clicks, favorites, or ratings. Percent shared is the average percentage of items users have in common with their friends. Data sources were chosen for their diversity of attributes.

Competing methods. We compared SPF to five competing models that involve a social network in recommendation: RSTE [24], TrustSVD [14], SocialMF [17], SoRec [25], and TrustMF [34].[6] We also include probabilistic Gaussian matrix factorization (PMF) [29], because it is a widely used recommendation method. For each of these, we used the parameter settings that achieved best performance according to the example fits published on the LibRec website.

We can think of SPF having two parts: a Poisson factorization component and a social component (see Eq. 1). Thus we also compared SPF to each of these components in isolation, Poisson factorization [11] (PF) and *social factorization* (SF). SF is the influence model without the factorization model.[7] We note that SF is a contribution of this paper.

Finally, we compare to two baselines, ordering items randomly and ordering items by their universal popularity.

Metrics. We evaluate these methods on a per-user basis. For each user, we predict clicks for both held-out and truly unclicked items, and we rank these items according to their predictions. We denote the user-specific rank to be rank_{ui} for item i and user u. A better model will place the held-out items higher in the ranking (giving smaller rank_{ui} values on held-out items). We now introduce the *normalized cumulative reciprocal rank* (NCRR) metric to gauge this performance.

Reciprocal rank (RR) is an information retrieval measure; given a query, it is the reciprocal of the rank at which the first relevant document was retrieved. (Larger numbers are better.) Users "query" a recommender system similarly, except that each user only has one query (e.g., "what books should I read?") and they care not just about the first item that's relevant, but about finding as many relevant items as possible.

Suppose user u has held out items $\mathcal{D}_u$.[8] We define the cumulative reciprocal rank to be:

$$\text{CRR}_u = \sum_{i \in \mathcal{D}_u} \frac{1}{\text{rank}_{ui}}.$$

CRR can be interpreted as the ease of finding all held-out items, as higher numbers indicate that the held-out items are higher in the list. For example, a CRR of 0.75 means that the second and fourth items are in the held-out set, or are relevant to the user.

[6]We used the LibRec library (librec.net) for all competing methods.

[7]Social factorization has a technical problem when none of a user's friends has clicked on an item; the resulting Poisson cannot have a rate of zero. Thus we add a small constant $\epsilon = 10^{-10}$ to the rate in social factorization's model of clicks.

[8]With binary data this is simply the full set of heldout items. When items have non-binary ratings, we threshold the set such to include only highly rated items (4 or 5 in a 5-star system).

CRR behaves similarly to discounted cumulative gain (DCG), except it places a higher priority on high-rank items by omitting the log factor—it can be thought of as a harsher variant of DCG. Like DCG, it can be also be normalized. The normalized cumulative reciprocal rank (NCRR) is

$$\text{NCRR}_u = \frac{\text{CRR}_u}{\text{ideal CRR}_u},$$

where the ideal variant in the denominator is the value of the metric if the ranking was perfect. To evaluate an entire model, we can compute average NCRR over all users, $\frac{1}{U}\sum_u \text{NCRR}_u$. We will use this metric throughout this section.

Performance measured by NCRR is consistent with performance measured by NDCG, but NCRR is more interpretable—simple reciprocals are easier to understand than the reciprocal of the log.

Note we omit root-mean-square error (RMSE) as a metric. Improvements in RMSE often do not translate into accuracy improvements for ranked lists [1, 7, 23, 31], especially with binary or implicit data. Our end goal here is item recommendation and not rating prediction—"which movie should I watch next?" is inherently a ranking problem—thus we treat the predictions as means to an end.

3.2 Performance and exploration

We evaluate SPF by considering overall performance and performance as a function of user degree. We also show how to explore the data using the algorithm.

Performance. Figure 3 shows the performance of SPF against the competing methods: the previous methods that account for the social network, social factorization (SF), Poisson factorization (PF), and the popularity baseline. (We do not illustrate the random baseline because it is far below all of the other methods.) SPF achieves top performance on five of the datasets. On the one remaining dataset, Etsy, the social-only variant of our model (SF) performs best.

Notice the strong performance of ranking by popularity. This highlights the importance of social factorization. It is only social Poisson factorization that consistently outperforms this baseline.

We measured runtime with the Ciao data set to get a sense for the relative computational costs. Figure 4 shows the runtime for all of the methods at various values of K. The Poisson models are average in terms of runtime.

Finally, using the Ciao and Epinions data, we break down the performance of SPF, SF, and PF as a function of the degree of each user; the results are shown in Figure 5.[9] All models perform better on high-degree users, presumably because these are higher activity users as well. Overall, SPF performs better than SF because of its advantage on the large number of low-degree users.

[9]Smoothed with GAM. http://www.inside-r.org/r-doc/mgcv/gam

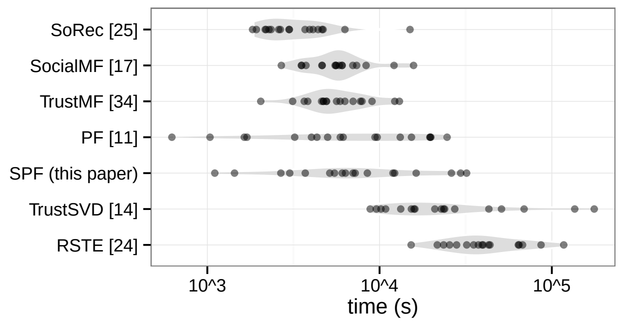

Figure 4: Training and testing runtimes for multiple models on Ciao data, with the number of latent factors K ranging from 1 to 500. Each dot represents a full cycle of training and evaluating. SPF performs with average runtime.

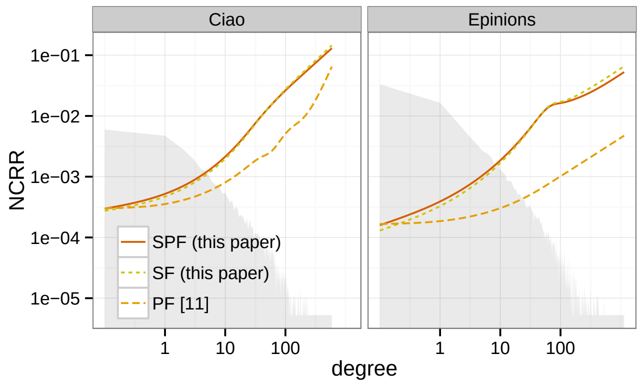

Figure 5: Performance on Ciao and Epinions broken down as a function of degree; grey in background indicates density of users. SPF and SF perform similarly, with SPF doing slightly better on a large number of low-degree users and SF doing better on a low number of high-degree users.

Interpretability. It is important to be able to explain the origins of recommendations to users [15]. Items recommended with SPF have the advantage of interpretability. In particular, we use auxiliary variables (see Appendix) to attribute each recommendation to friends or general preferences; we then use these attributions to explore data.

When items are recommended because of social influence, the system may indicate a friend as the source of the recommendation. Similarly, when items are recommended because of general preferences, the system may indicate already clicked items that exhibit that preference. On the Etsy data, learned item factors included coherent groupings of items such as mugs, sparkly nail polish, children's toys, handmade cards, and doll clothes. Thus, SPF explains the recommended the handmade soap in Figure 1 as coming from general preferences and the others items as coming from social influence. The social and preference signals will not always be cleanly separated; SPF attributes recommendations to sources probabilistically.

Figure 6 shows how the proportion of social attribution (as opposed to general preference attribution) changes as a function of user degree on Ciao and Epinions. We observe that Epinions attributes a larger portion of behavior to social influence, controlled for user degree. Similarly, we can compute the contribution of users to their friends' behavior. Figure 7 shows social contribution as a function of indegree; here we see that Epinions users with higher indegree have lower social contribution than low-indegree users.

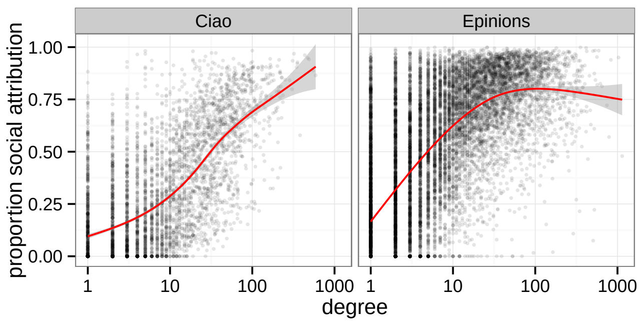

Figure 6: The proportion of social attribution (vs. general preference attribution) as a function of user degree. Attributions are calculated on all training data from Ciao and Epinions. Epinions attributes a larger portion of rating to social influence.

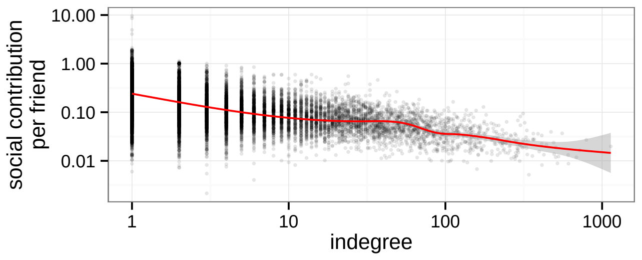

Figure 7: Contribution to friends' behavior as a function of indegree, calculated on all Epinions training data. Users with higher indegree have lower social contribution.

3.3 Experimental details

The details of our methods requires some decisions: we must choose the number of latent factors K and set the hyperparameters.

Choosing the number of latent factors K. All factorization models, including SPF, require the investigator to select of the number of latent factors K used to represent users and items. We evaluated the sensitivity to this choice for the Ciao dataset. (We chose this dataset because of its smaller size; ranking millions of items for every user is computationally expensive for any model.) Figure 8 shows per-user average NCRR K varies from 1 to 500; SPF performs best on the Ciao dataset with $K = 40$, though is less sensitive to this choice than some other methods (such as PF).

Hyperparameters. We also must set the hyperparameters to the gamma priors on the latent variables. The gamma is parameterized by a shape and a rate. We followed [11] and set them to 0.3 for the priors on latent preferences and attributes. We set the hyperparameters for the prior on user influences to $(2, 5)$ in order to encourage the model to explore explanation by social influence. In a pilot study, we found that the model was not sensitive to these settings.

Does learning influence matter? We can easily fix each user-friend influence at 1, giving us local popularity among a user's social connections. We compared fitted influence against fixed influence on both Ciao and Epinions and found that SPF with fitted influence performs best on both datasets.

In the case of cold-start users, where we know the user's social network but not their click counts on items, SPF will perform equivalently to SF with fixed influence. SPF in this cold-start user scenario performs better than competing models.

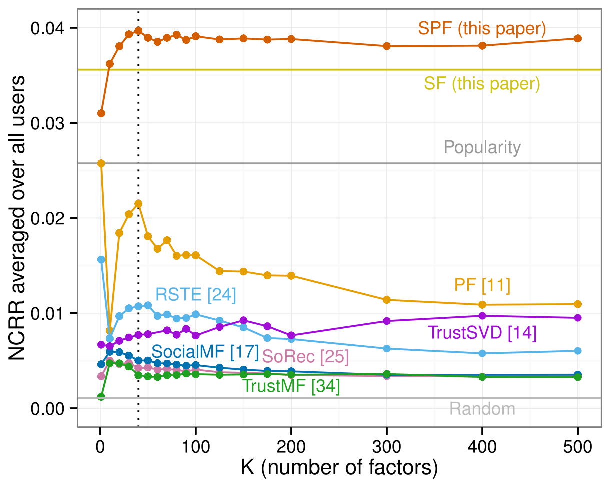

Figure 8: Model performance on Ciao data (measured as NCRR averaged over all users) as a function of number of latent factors K. The dotted vertical line at $K = 40$ indicates the best performance for Poisson family models.

4. DISCUSSION

We presented social Poisson factorization, a Bayesian model that incorporates a user's latent preferences for items with the latent influences of her friends. We demonstrated that social Poisson factorization improves recommendations even with noisy online social signals. Social Poisson factorization has the following properties: (1) It discovers the latent influence that exists between users in a social network, allowing us to analyze the social dynamics. (2) It provides a source of explainable serendipity (i.e., pleasant surprise due to novelty). (3) It enjoys scalable algorithms that can be fit to large data sets.

We anticipate that social Poisson factorization will perform well on platforms that allow for and encourage users to share content. Examples include Etsy, Pinterest, Twitter, and Facebook. We note that our model does not account for time—when two connected users both enjoy an item, one of them probably consumed it first. Future work includes incorporating time, hierarchical influence, and topical influence.

5. ACKNOWLEDGMENTS

We thank Prem Gopalan, Jake Hofman, Chong Wang, Laurent Charlin, Rajesh Ranganath, and Alp Kucukelbir for their insights and discussions. We thank Etsy, Diane Hu in particular, for sharing data. We also thank LibRec's creator, Guibing Guo. DMB is supported by NSF BIGDATA NSF IIS-1247664, ONR N00014-11-1-0651, and DARPA FA8750-14-2-0009. TER is supported by NSF CNS-1314603, by DTRA HDTRA1-10-1-0120, and by DAPRA under SMISC Program Agreement No. W911NF-12-C-0028.

6. REFERENCES

[1] X. Amatriain, P. Castells, A. de Vries, and C. Posse. Workshop on recommendation utility evaluation: Beyond RMSE. In *RecSys*, pages 351–352, 2012.

[2] R. Andersen, C. Borgs, J. Chayes, U. Feige, A. Flaxman, A. Kalai, V. Mirrokni, and M. Tennenholtz. Trust-based recommendation systems: an axiomatic approach. In *WWW*, pages 199–208, 2008.

[3] B. C. Arnold, E. Castillo, and J. M. Sarabia. *Conditional specification of statistical models*. Springer, 1999.

[4] C. Bishop. *Pattern Recognition and Machine Learning*. Springer New York, 2006.

[5] J. Canny. GaP: a factor model for discrete data. In *SIGIR*, pages 122–129, 2004.

[6] D. Crandall, D. Cosley, D. Huttenlocher, J. Kleinberg, and S. Suri. Feedback effects between similarity and social influence in online communities. In *KDD*, pages 160–168, 2008.

[7] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *RecSys*, pages 39–46, 2010.

[8] N. Du, L. Song, H. Woo, and H. Zha. Uncover topic-sensitive information diffusion networks. In *AISTATS*, pages 229–237, 2013.

[9] S. Fazeli, B. Loni, A. Bellogin, H. Drachsler, and P. Sloep. Implicit vs. explicit trust in social matrix factorization. In *RecSys*, pages 317–320, 2014.

[10] J. Golbeck and J. Hendler. FilmTrust: Movie recommendations using trust in web-based social networks. *TOIT*, 6(4):497–529, Jan. 2006.

[11] P. Gopalan, J. M. Hofman, and D. M. Blei. Scalable recommendation with hierarchical Poisson factorization. In *UAI*, pages 326–335, 2015.

[12] A. Guille, H. Hacid, C. Favre, and D. A. Zighed. Information diffusion in online social networks: A survey. *SIGMOD Record*, 42(2):17–28, July 2013.

[13] G. Guo, J. Zhang, D. Thalmann, and N. Yorke-Smith. Etaf: An extended trust antecedents framework for trust prediction. In *ASONAM*, pages 540–547, 2014.

[14] G. Guo, J. Zhang, and N. Yorke-Smith. TrustSVD: Collaborative filtering with both the explicit and implicit influence of user trust and of item ratings. *AAAI*, pages 123–129, 2015.

[15] J. L. Herlocker, J. A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *CSCW*, pages 241–250, 2000.

[16] M. Hoffman, D. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1303–1347), 2013.

[17] M. Jamali and M. Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *RecSys*, pages 135–142, 2010.

[18] J. Johnstone and E. Katz. Youth and popular music: A study in the sociology of taste. *Journal of Sociology*, 62(6):563–568, May 1957.

[19] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, Nov. 1999.

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

[21] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562, 2000.

[22] J. Leskovec, A. Singh, and J. Kleinberg. Patterns of influence in a recommendation network. In *PAKDD*, pages 380–389, 2006.

[23] D. Loiacono, A. Lommatzsch, and R. Turrin. An analysis of the 2014 recsys challenge. In *RecSysChallenge*, page 1, 2014.

[24] H. Ma, I. King, and M. R. Lyu. Learning to recommend with social trust ensemble. In *SIGIR*, pages 203–210, 2009.

[25] H. Ma, H. Yang, M. R. Lyu, and I. King. SoRec: Social recommendation using probabilistic matrix factorization. In *CIKM*, pages 931–940, 2008.

[26] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King. Recommender systems with social regularization. In *WSDM*, pages 287–296, 2011.

[27] P. Massa and P. Avesani. Trust-aware recommender systems. In *RecSys*, pages 17–24, 2007.

[28] S. Purushotham, Y. Liu, and C.-C. J. Kuo. Collaborative topic regression with social matrix factorization for recommendation systems. *CoRR*, abs/1206.4684, 2012.

[29] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *NIPS*, pages 1257–1264, 2007.

[30] S. Shang, P. Hui, S. R. Kulkarni, and P. W. Cuff. Wisdom of the crowd: Incorporating social influence in recommendation models. In *ICPADS*, pages 835–840, 2011.

[31] P. Singh, G. Singh, and A. Bhardwaj. Ranking approach to recsys challenge. In *RecSysChallenge*, page 19, 2014.

[32] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, page 4, Jan. 2009.

[33] I. P. Volz. The impact of online music services on the demand for stars in the music industry. In *WWW*, pages 659–667, 2006.

[34] B. Yang, Y. Lei, D. Liu, and J. Liu. Social collaborative filtering by trust. In *IJCAI*, pages 2747–2753, 2013.

[35] M. Ye, X. Liu, and W.-C. Lee. Exploring social influence for recommendation: A generative model approach. In *SIGIR*, pages 671–680, 2012.

[36] T. Zhao, J. McAuley, and I. King. Leveraging social connections to improve personalized ranking for collaborative filtering. In *CIKM*, pages 261–270, 2014.

APPENDIX

In this appendix, we describe the details of the variational inference algorithm for SPF. This algorithm fits the parameters of the variational distribution in Eq. 3 so that it is close in KL divergence to the posterior. We use coordinate ascent, iteratively updating each parameter while holding the others fixed. This goes uphill in the variational objective and converges to a local optimum [4].

To obtain simple updates, we first construct auxiliary latent variables z. These variables, when marginalized out, leave the original model intact. Recall the additive property of the Poisson distribution. Specifically, if $r \sim \text{Poisson}(a+b)$ then $r = z_1 + z_2$, where $z_1 \sim \text{Poisson}(a)$ and $z_2 \sim \text{Poisson}(b)$. We apply this decomposition to the conditional click count distribution in Eq. 1. We define Poisson variables for each term in the click count:

$$z^M_{uik} \sim \text{Poisson}(\theta_{uk}\beta_{ik}) \quad z^S_{uiv} \sim \text{Poisson}(\tau_{uv} r_{vi}).$$

The M and S superscripts indicate the contributions from matrix factorization (general preferences) and social factorization (influence), respectively. Given these variables, the click count is deterministic,

$$r_{ui} \mid r_{-u,i} = \textstyle\sum_{k=1}^K z^M_{uik} + \sum_{v=1}^V z^S_{uiv},$$

where $V = |N(u)|$ and the index v selects a friend of u (as opposed to selecting from the set of all users).

Coordinate-ascent variational inference is derived from the complete conditionals, i.e., the conditional distributions of each variable given the other variables and observations. These conditionals define both the form of each variational factor and their updates. For the Gamma variables—the user preferences, item attributes, and user influence—the conditionals are

$$\theta_{uk} \mid \beta, \tau, z, \mathbf{R}, \mathbf{N} \sim \text{Gam}\left(a_\theta + \sum_i z^M_{uik},\ b_\theta + \sum_i \beta_{ik}\right) \tag{5}$$

$$\beta_{ik} \mid \theta, \tau, z, \mathbf{R}, \mathbf{N} \sim \text{Gam}\left(a_\beta + \sum_u z^M_{uik},\ b_\beta + \sum_u \theta_{uk}\right) \tag{6}$$

$$\tau_{uv} \mid \theta, \beta, z, \mathbf{R}, \mathbf{N} \sim \text{Gam}\left(a_\tau + \sum_i z^S_{uiv},\ b_\tau + \sum_i r_{vi}\right). \tag{7}$$

The complete conditional for the auxiliary variables is $z_{ui} \mid \theta, \beta, \tau, \mathbf{R}, \mathbf{N} \sim \text{Mult}(r_{ui}, \phi_{ui})$ where

$$\phi_{ui} \propto \left\langle \theta_{u1}\beta_{i1}, \cdots, \theta_{uK}\beta_{iK}, \tau_{u1}r_{1i}, \cdots, \tau_{uV}r_{Vi} \right\rangle. \tag{8}$$

(Intuitively, these variables allocate the data to one of the factors or one of the friends.) Each variational factor is set to the same family as its corresponding complete conditional.

Given these conditionals, the algorithm sets each parameter to the expected conditional parameter under the variational distribution. (Thanks to the mean field assumption, this expectation will not involve the parameter being updated.) Note that under a gamma distribution, $\mathrm{E}[\lambda] = \lambda_a/\lambda_b$, where λ_a and λ_b are shape and rate parameters. For the auxiliary variables, the expectation of the indicator is the probability, $\mathrm{E}[z_{ui}] = r_{ui} * \phi_{ui}$.

Algorithm 1 shows our variational inference algorithm. It is $O(N(K+V))$ per iteration, where N is the number of recorded user-item interactions (click counts, ratings, etc.). K is the number of latent factors, and V is the maximum user degree. (Note that both K and V are usually small relative to N.) We can modify the algorithm to sample users and update the variables stochastically [16]; this approach scales to much larger datasets than competing methods.

Algorithm 1 Mean field variational inference SPF

```
 1: initialize $\mathrm{E}[\theta]$, $\mathrm{E}[\beta]$ randomly
 2: for each user $u$ do
 3:     for each friend $v \in N(u)$ do
 4:         $\lambda^{\tau,b}_{u,v} = \text{prior } b_\tau + \sum_i r_{vi}$                ▷ see Eq. 7
 5: while $\Delta \log \mathcal{L} > \delta$ do                ▷ check for model convergence
 6:     init. global $\lambda^{\beta,a}$ to prior $a_\beta$ for all items and all factors
 7:     for each user $u$ do
 8:         while $\Delta[\theta_u] + \Delta[\tau_u] > \delta'$ do                ▷ user convergence
 9:             init. local $\lambda^{\beta,a}$ to 0 for all items and factors
10:             init. preferences $\lambda^{\theta,a}_u$ to prior $a_\theta$ for all factors
11:             $\lambda^{\theta,b}_u = \text{prior } b_\theta + \sum_i \mathrm{E}[\beta_i]$                ▷ see Eq. 5
12:             init. influence $\lambda^{\tau,a}_{user}$ to prior $a_\tau$ for all friends
13:             for each (item $i$, click count $r$) $\in clicks_u$ do
14:                 set $\phi_{ui}$ from $\mathrm{E}[\theta_u]$, $\mathrm{E}[\beta_i]$, $\mathrm{E}[\tau_u]$, and $r_i$ (Eq. 8)
15:                 $\mathrm{E}[z_{ui}] = r * \phi_{ui}$
16:                 update $\lambda^{\theta,a}_u$ += $\mathrm{E}[z^M_{ui}]$                ▷ see Eq. 5
17:                 update $\lambda^{\tau,a}_u$ += $\mathrm{E}[z^S_{ui}]$                ▷ see Eq. 7
18:                 update local $\lambda^{\beta,a}_i$ += $\mathrm{E}[z^M_{ui}]$                ▷ see Eq. 6
19:             $\mathrm{E}[\theta_u] = \lambda^{\theta,a}_u / \lambda^{\theta,b}_u$
20:             $\mathrm{E}[\tau_u] = \lambda^{\tau,a}_u / \lambda^{\tau,b}_u$
21:         global $\lambda^{\beta,a}$ += local $\lambda^{\beta,a}$
22:     $\lambda^{\beta,b} = \text{prior } b_\beta + \sum_u \mathrm{E}[\theta_u]$                ▷ see Eq. 6
23:     $\mathrm{E}[\beta] = \lambda^{\beta,a} / \lambda^{\beta,b}$
```

To assess convergence, we use the change in the average click log likelihood of a validation set.