Scalable Recommendation with Poisson Factorization

Prem Gopalan
Princeton University
35 Olden Street
Princeton, NJ
pgopalan@cs.princeton.edu

Jake M. Hofman
Microsoft Research
641 Sixth Avenue, Floor 7
New York, NY
jmh@microsoft.com

David M. Blei
Princeton University
35 Olden Street
Princeton, NJ
blei@cs.princeton.edu

ABSTRACT

We develop hierarchical Poisson matrix factorization (HPF) for recommendation. HPF models sparse user behavior data, large user/item matrices where each user has provided feedback on only a small subset of items. HPF handles both explicit ratings, such as a number of stars, or implicit ratings, such as views, clicks, or purchases. We develop a variational algorithm for approximate posterior inference that scales up to massive data sets, and we demonstrate its performance on a wide variety of real-world recommendation problems–users rating movies, users listening to songs, users reading scientific papers, and users reading news articles. Our study reveals that hierarchical Poisson factorization definitively outperforms previous methods, including nonnegative matrix factorization, topic models, and probabilistic matrix factorization techniques.

1. INTRODUCTION

Recommendation systems are a vital component of the modern Web. They help readers effectively navigate otherwise unwieldy archives of information and help websites direct users to items—movies, articles, songs, products—that they will like. A recommendation system is built from user behavior data, historical data about which items each user has consumed, be it clicked, viewed, rated, or purchased. First, we uncover the behavioral patterns that characterize various types of users and the kinds of items they tend to like. Then, we exploit these discovered patterns to recommend future items to its users.

In this paper, we develop Poisson factorization (PF) algorithms for recommendation. Our algorithms easily scale to massive data and significantly outperform the existing methods. We show that Poisson factorization for recommendation is tailored to real-world properties of user behavior data: the heterogenous interests of users, the varied types of items, and a realistic distribution of the finite resources that users have to consume items.

Figure 1 illustrates Poisson factorization on data from Netflix. The Netflix data contains the ratings of 480,000 users on 17,000 movies, organized in a matrix of 8.16B cells (and containing 250M ratings). From these data, we extract the patterns of users' interests and the movies that are associated with those interests. The left panel illustrates some of those patterns—the algorithm has uncovered action movies, independent comedies, and 1980s science fiction.

The top panel illustrates how we can use these patterns to form recommendations for an (imaginary) user. This user enjoys various types of movies, including fantasy ("Lord of the Rings"), classic science fiction ("Star Wars: Episode V"), and independent comedies ("Clerks", "High Fidelity"). Of course, she has only seen a handful of the available movies. PF first uses the movies she has seen to infer what kinds of movies she is interested in, and then uses these inferred interests to suggest new movies. The list of movies at the bottom of the figure was suggested by our algorithm. It includes other comedies (such as "The Big Lebowski") and other science fiction (such as "Star Wars: Episode II").

In more detail, Poisson factorization is a probabilistic model of users and items. It associates each user with a latent vector of preferences, each item with a latent vector of attributes, and constrains both sets of vectors to be sparse and non-negative. Each cell of the observed behavior matrix is assumed drawn from a Poisson distribution—an exponential family distribution over non-negative integers—whose parameter is a linear combination of the corresponding user preferences and item attributes. The main computational problem is posterior inference: given an observed matrix of user behavior, we discover the latent attributes that describe the items and the latent preferences of the users. For example, the components in Figure 1 (left) illustrate the top items for specific attribute dimensions and the plot in Figure 1 (middle) illustrates the estimated preference vector for the given user. A spike in the preference vector implies that the user tends to like items with the corresponding latent attribute.

This general procedure is common to many variants of matrix factorization. We found, however, that PF enjoys significant quantitative advantages over classical methods and for a wide variety of data sets, including those with implicit feedback (a binary matrix indicating which items users consumed) and those with explicit feedback (a matrix of integer ratings). Figure 4 shows that PF, and its hierarchical variant HPF, perform significantly better than existing methods—including the industry standard of matrix factorization with user and item biases (MF)—for large data sets of Netflix users watching movies, Last.FM users

"Action"	"Indie Comedy, Romance"	"80's Science Fiction"
The Matrix	Grosse Pointe Blank	Star Wars: Episode IV: A New Hope
The Matrix: Reloaded	Four Weddings and a Funeral	Star Wars: Episode VI: Return of the Jedi
Spider-Man	High Fidelity	Star Wars: Episode V: The Empire Strikes Back
X2: X-Men United	Much Ado About Nothing	Back to the Future Part II

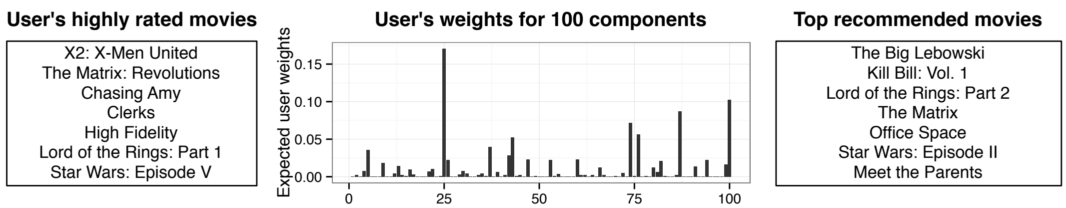

Figure 1: The top panel shows the top movies in 3 components for a user from the Netflix data set. The bottom panel is an illustration showing a subset of the highly rated movies by this user, and the right panel shows movies recommended to the user by our algorithm. The expected user's K-vector of weights θ_u, inferred by our algorithm is shown in the middle panel.

listening to music, scientists reading papers, and *New York Times* readers clicking on articles.

There are two main advantages of Poisson factorization over traditional methods, both of which contribute to its superior empirical performance. First, it better captures real consumption data, specifically that users have finite (and varied) resources with which to view items. To see this, we can rewrite the model as a two stage process where a user first decides on a budget of movies to watch and then spends this budget watching movies that she is interested in. If the model accurately captures the distribution of budgets then watched items carry more weight than unwatched items, because unwatched items can be partially explained by a lack of resources. We conjecture that classical matrix factorization systematically overestimates the users' budgets, and we confirm this hypothesis in Section 4 using a posterior predictive check [8]. This misfit leads to an overweighting of the zeros, which explains why practitioners require complex methods for downweighting them [14, 7, 6, 26]. Poisson factorization does not need to be modified in this way.

The second advantage of PF algorithms is that they need only iterate over the viewed items in the observed matrix of user behavior, i.e., the non-zero elements, and this is true even for implicit or "positive only" data sets. (This follows from the mathematical form of the Poisson distribution.) Thus, Poisson factorization takes advantage of the natural sparsity of user behavior data and can easily analyze massive real-world data. In contrast, classical matrix factorization based on the Gaussian distribution [29] must iterate over both positive and negative examples in the implicit setting. Thus it cannot take advantage of data sparsity, which makes computation difficult for even modestly sized problems. For example, one cannot fit to the full Netflix data set (as we did in Figure 1) without appealing to stochastic optimization [23]. We note that our algorithms are also amenable to stochastic optimization, which we can use to analyze data sets even larger than those we studied.

We review related work below before discussing details of the Poisson factorization model, including its statistical properties and methods for scalable inference.

2. RELATED WORK

The roots of Poisson factorization come from nonnegative matrix factorization [21], where the objective function is equivalent to a factorized Poisson likelihood. The original NMF update equations have been shown to be an expectation-maximization (EM) algorithm for maximum likelihood estimation of a Poisson model via data augmentation [5].

Placing a Gamma prior on the user weights results in the GaP model [4], which was developed as an alternative text model to latent Dirichlet allocation (LDA) [3, 15]. The GaP model is fit using the expectation-maximization algorithm to obtain point estimates for user preferences and item attributes. The Probabilistic Factor Model (PFM) [22] improves upon GaP by placing a Gamma prior on the item weights as well, and using multiplicative update rules to infer an approximate maximum a posteriori estimate of the latent factors. In contrast, as explained below, our model uses a hierarchical prior structure of Gamma priors on user and item weights, and Gamma priors over the rate parameters from which these weights are drawn. This enables us to accurately model the skew in user activity and item popularity, which contributes to good predictive performance. Furthermore, we approximate the full posterior over all latent factors using a scalable variational inference algorithm.

Independently of GaP and user behavior models, Poisson factorization has been studied in the context of signal processing for source separation [5, 11] and for the purpose of detecting community structure in network data [1, 10]. This research includes variational approximations to the posterior, though the issues and details around these data differ significantly from user data we consider and our derivation below (based on auxiliary variables) is more direct.

When modeling implicit feedback data sets, researchers have proposed merging factorization techniques with neighborhood models [19], weighting techniques to adjust the relative importance of positive examples [14], and sampling-based approaches to create informative negative examples [7, 6, 26]. In addition to the difficulty in appropriately weighting or sampling negative examples, there is a known selection bias in provided ratings that causes further complications [25, 24]. Poisson factorization does not require such special adjustments and scales linearly with the number of observed ratings.

We discuss additional related recommendation methods in Section 4, where we compare a variety of applicable methods to Poisson factorization empirically.

3. POISSON RECOMMENDATION

In this section we describe the Poisson factorization model for recommendation, and discuss its statistical properties.

We are given data about users and items, where each user has consumed and possibly rated a set of items. The observation y_{ui} is the rating that user u gave to item i, or zero if no rating was given. (In so-called "implicit" consumer data, y_{ui} equals one if user u consumed item i and zero otherwise.) User behavior data, such as purchases, ratings, clicks, or views, are typically sparse. Most of the values of the matrix y are zero.

We model the data with factorized Poisson distributions [4]. We represent each item i as a vector of K latent attributes β_i and represent each user u as a vector of K latent preferences β_u. These vectors are sparse and non-negative. The observations y_{ui} are modeled with a Poisson, parameterized by the inner product of the user preferences and item attributes, $y_{ui} \sim \text{Poisson}(\theta_u^\top \beta_i)$. This is a variant of probabilistic matrix factorization [30] but where each user and item's weights are positive [21] and where the Poisson replaces the Gaussian.

Beyond the basic data generating distribution, we place Gamma priors on the latent attributes and latent preferences, which encourage the model towards sparse representations of the users and items. Furthermore, we place additional priors on the user and item-specific rate parameter of those Gammas, which controls the average size of the representation. This hierarchical structure allows us to capture the diversity of users, some tending to consume more than others, and the diversity of items, some being more popular than others. The literature on recommendation systems suggests that a good model must capture such heterogeneity across users and items [20].

Putting this together, the generative process of the hierarchical Poisson factorization model (HPF) is as follows:

1. For each user u:
   (a) Sample activity $\xi_u \sim \text{Gamma}(a', a'/b')$.
   (b) For each component k, sample preference
   $$\theta_{uk} \sim \text{Gamma}(a, \xi_u).$$
2. For each item i:
   (a) Sample popularity $\eta_i \sim \text{Gamma}(c', c'/d')$.
   (b) For each component k, sample attribute
   $$\beta_{ik} \sim \text{Gamma}(c, \eta_i).$$
3. For each user u and item i, sample rating
   $$y_{ui} \sim \text{Poisson}(\theta_u^\top \beta_i).$$

This process describes the statistical assumptions behind the model. Note that we also study a sub-class of HPF where we fix the rate parameters for all users and items to the same pair of hyperparameters. We call this model Bayesian Poisson Factorization (BPF).

The central computational problem is posterior inference, which is akin to "reversing" the generative process. Given a user behavior matrix, we want to estimate the conditional distribution of the latent per-user and per-item structure, $p(\theta_{1:N}\beta_{1:M} \mid y)$. The posterior is the key to recommendation. We estimate the posterior expectation of each user's preferences, each items attributes and, subsequently, form predictions about which unconsumed items each user will like. We discuss our posterior inference algorithm in detail in Section 3.2. Figure 1 illustrates posterior estimates of β and θ on Netflix data.

Once the posterior is fit, we use HPF to recommend items to users by predicting which of the unconsumed items each will like. We rank each user's unconsumed items by their posterior expected Poisson parameters,

$$\text{score}_{ui} = \text{E}[\theta_u^\top \beta_i \mid y]. \tag{1}$$

This amounts to asking the model to rank by probability which of the presently unconsumed items each user will likely consume in the future.

3.1 Properties of HPF

With the modeling details in place, we highlight several statistical properties of hierarchical Poisson factorization. These properties provide advantages over classical (Gaussian) matrix factorization.[1]

HPF captures sparse factors. As we mentioned above, the Gamma priors on preferences and attributes encourages sparse representations of users and items. Specifically, by setting the shape parameter to be small, most of the weights will be close to zero and only a few will be large.

HPF models the long-tail of users and items. One statistical characteristic of real-world user behavior data is the distribution of user activity (i.e., how many items a user consumed) and item popularity (i.e., how many users consumed an item). These distributions tend to be long-tailed: while most users consume a handful few items, a few "tail users" consume thousands of items. A question we can ask of a statistical model of user behavior data is how well it captures these distributions. We found that HPF captures them very well, while classical matrix factorization does not.

To check this, we implemented a *posterior predictive check* (PPC) [28, 8], a technique for model assessment from the Bayesian statistics literature. The idea behind a PPC is to simulate a complete data set from the posterior predictive distribution—the distribution over data that the posterior induces—and then compare the generated data set to the true observations. A good model will produce data that captures the important characteristics of the observed data.

We developed a PPC for matrix factorization algorithms on user behavior data. First, we formed posterior estimates of user preferences and item attributes for both classical MF and HPF. Then, from these estimates, we generated a new user behavior matrix by drawing values for each user and item. (For classical matrix factorization, we truncated these values at zero and rounded to one in order to generate a plausible matrix.) Finally, we compared the matrix generated by the posterior predictive distribution to the true observations.

Figure 2 illustrates our PPC for the Netflix data. In this figure, we illustrate three distributions over user activity:

[1]Specifically, by classical matrix factorization we mean L2 regularized matrix factorization with bias terms for users and items, fit using stochastic gradient descent [20]. Without the bias terms, this corresponds to maximum a-posteriori inference under Probabilistic Matrix Factorization [30].

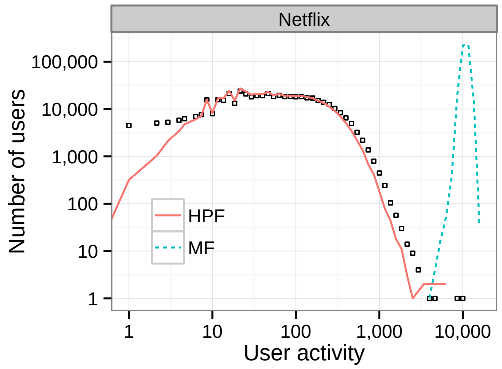

Figure 2: A posterior predictive check of the distribution of total ratings for the Netflix data set. The pink curve shows the empirical count of the number of users who have rated a given number of items, while the green and blue curves show the simulated totals from fitted Poisson and traditional matrix factorization models, respectively. The Poisson marginal closely matches the empirical, whereas classical matrix factorization fits a large mean to account for skew in the distribution and the missing ratings.

the observed distribution (squares), the distribution from a data set replicated by HPF (red line), and a distribution from a data set replicated by classical MF (blue line). HPF captures the truth much more closely than classical MF, which badly overestimates the distribution of user activity. (We note that this is true for the item popularities as well, and for the other data sets.) This indicates that HPF better represents real data when measured by its ability to capture distributions of user activity and item popularity.

HPF downweights the effect of zeros. Another advantage of HPF is that it implicitly down-weights the contribution of the items that each user did not consume. With an appropriate fit to user activity, the model has two ways of explaining an unconsumed item: either the user is not interested in it or she would be interested in it but is likely to not be further active. In contrast, a user that consumes an item must be interested in it. Thus, the model benefits more from making a consumed user/item pair more similar than making an unconsumed user/item pair less similar.

Classical MF is based on Gaussian likelihoods (i.e., squared loss), which gives equal weight to consumed and unconsumed items. Consequently, when faced with a sparse matrix and implicit feedback, i.e., binary consumption data, matrix factorization places more total emphasis on the unconsumed user/item pairs. (This too can be seen to stem from classical MF's overestimation of the distribution of user activity.) To address this, researchers have patched the model in complex ways, for example, by including per-observation confidences [20] or considering all zeroes to be hidden variables [26]. Poisson factorization more naturally solves this problem by better capturing each user's rate of consumption.

As an example, consider two similar science fiction movies, "Star Wars" and "The Empire Strikes Back", and consider a user who has seen one of them. The Gaussian model pays an equal penalty for making the user similar to these items as it does for making the user different from them—with quadratic loss, seeing "Star Wars" is evidence for liking science fiction, but not seeing "The Empire Strikes Back" is evidence for disliking it. The Poisson model, however, will prefer to bring the user's latent weights closer to the movies' weights because it favors the information from the user watching "Star Wars". Further, because the movies are similar, this increases the Poisson model's predictive score that a user who watches "Star Wars" will also watch "The Empire Strikes Back".

Fast inference with sparse matrices. Finally, the likelihood of the observed data under HPF (and BPF) depends only on the consumed items, that is, the non-zero elements of the user/item matrix y. This facilates computation for the kind of sparse matrices that we tend to observe in real-world data.

We can see this property from the form of the Poisson distribution. Given the latent preferences θ_u and latent attributes β_i, the Poisson distribution of the rating y_{ui} is

$$p(y_{ui} \mid \theta_u, \beta_i) = \left(\theta_u^\top \beta_i\right)^y \exp\left\{-\theta_u^\top \beta_i\right\} / y_{ui}! \qquad (2)$$

Recall the elementary fact that $0! = 1$. The log probability of the complete matrix y is

$$\log p(y \mid \theta, \beta) = \left(\sum_{\{y_{ui}>0\}} y_{ui} \log(\theta_u^\top \beta_i) - \log y_{ui}!\right) \qquad (3)$$
$$- \left(\sum_u \theta_u\right)^\top \left(\sum_i \beta_i\right).$$

Classical MF does not enjoy this property. These methods, especially when applied to massive data sets of implicit feedback, must (in theory) iterate over all the cells of the matrix. Practitioners require solutions such as sub-sampling [6] approximation [14], or stochastic optimization [23].

3.2 Inference with variational methods

Using HPF for recommendation hinges on solving the posterior inference problem. Given a set of observed ratings, we would like to infer the user preferences and item attributes that explain these ratings, and then use these inferences to recommend new content to the users. In this section we discuss the details and practical challenges of posterior inference for HPF, and present a mean-field variational inference algorithm as a practical and scalable approach. Our algorithm easily accommodates data sets with millions of users and hundreds of thousands of items on a single CPU.

Given an observed matrix of user behavior y, we would like to compute the posterior distribution of user preferences θ_{uk}, item attributes β_{ik}, user activity ξ_u and item popularity η_i. As for many Bayesian models of interest, however, the posterior is intractable to compute exactly. We show how to efficiently approximate the posterior with mean-field variational inference.

Variational inference is an optimization-based strategy for approximating posterior distributions in complex probabilistic models [18, 31]. Variational inference algorithms posit a family of distributions over the hidden variables, indexed by free "variational" parameters, and then finds the member of that family that is closest in Kullback-Liebler (KL) divergence to the true posterior. (The form of the family is chosen to make this optimization possible.) Thus, variational inference turns the inference problem into an optimization problem. Variational inference tends to scale better than alternative sampling-based approaches, like Monte Carlo Markov chain sampling, and has been deployed to solve many applied problems with complex models, including large-scale recommendation [26].

We will describe a simple variational inference algorithm for HPF. To do so, however, we first give an alternative formulation of the model in which we add an additional layer of latent variables. These auxiliary variables allow us to take advantage of some general results for variational algorithms [9, 13].

For each user and item we add K latent variables $z_{uik} \sim \text{Poisson}(\theta_{uk}\beta_{ik})$, which are integers that sum to the user/item value y_{ui}. A sum of Poisson random variables is itself a Poisson with rate equal to the sum of the rates. Thus, these new latent variables preserve the marginal distribution of the observation, $y_{ui} \sim \text{Poisson}(\theta_u^\top \beta_i)$. These variables can be thought of as the contribution from component k to the total observation y_{ui}. Note that when $y_{ui} = 0$, these auxiliary variables are not random—the posterior distribution of z_{ui} will place all its mass on the zero vector. Consequently, our inference procedure need only consider z_{ui} for those user/item pairs where $y_{ui} > 0$.

For all users and items, initialize the user parameters γ_u, κ_u^{rte} and item parameters λ_i, τ_i^{rte} to the prior with a small random offset. Set the user activity and item popularity shape parameters:

$$\kappa_u^{\text{shp}} = a' + Ka; \quad \tau_i^{\text{shp}} = c' + Kc$$

Repeat until convergence:

1. For each user/item such that $y_{ui} > 0$, update the multinomial:

$$\phi_{ui} \propto \exp\{\Psi(\gamma_{uk}^{\text{shp}}) - \log \gamma_{uk}^{\text{rte}} + \Psi(\lambda_{ik}^{\text{shp}}) - \log \lambda_{ik}^{\text{rte}}\}.$$

2. For each user, update the user weight and activity parameters:

$$\gamma_{uk}^{\text{shp}} = a + \textstyle\sum_i y_{ui}\phi_{uik}$$
$$\gamma_{uk}^{\text{rte}} = \frac{\kappa_u^{\text{shp}}}{\kappa_u^{\text{rte}}} + \textstyle\sum_i \lambda_{ik}^{\text{shp}}/\lambda_{ik}^{\text{rte}}$$
$$\kappa_u^{\text{rte}} = \frac{a'}{b'} + \sum_k \frac{\gamma_{uk}^{\text{shp}}}{\gamma_{uk}^{\text{rte}}}$$

3. For each item, update the item weight and popularity parameters:

$$\lambda_{ik}^{\text{shp}} = c + \textstyle\sum_u y_{ui}\phi_{uik}$$
$$\lambda_{ik}^{\text{rte}} = \frac{\tau_i^{\text{shp}}}{\tau_i^{\text{rte}}} + \textstyle\sum_u \gamma_{uk}^{\text{shp}}/\gamma_{uk}^{\text{rte}}$$
$$\tau_i^{\text{rte}} = \frac{c'}{d'} + \sum_k \frac{\lambda_{ik}^{\text{shp}}}{\lambda_{ik}^{\text{rte}}}$$

Figure 3: Variational inference for Poisson factorization. Each iteration only needs to consider the non-zero elements of the user/item matrix.

With these latent variables in place, we now describe the algorithm. First, we posit the variational family over the hidden variables. Then we show how to optimize its parameters to find the member close to the posterior of interest.

The latent variables in the model are user weights θ_{uk}, item weights β_{ik}, and user-item contributions z_{uik}, which we represent as a K-vector of counts z_{ui}. The *mean-field family* considers these variables to be independent and each governed by its own distribution,

$$q(\beta, \theta, \xi, \eta, z) = \prod_{i,k} q(\beta_{ik} \mid \lambda_{ik}) \prod_{u,k} q(\theta_{uk} \mid \gamma_{uk}) \prod_u q(\xi_u \mid \kappa_u) \prod_i q(\eta_i \mid \tau_i) \prod_{u,i} q(z_{ui} \mid \phi_{ui}). \tag{4}$$

Though the variables are independent, this is a flexible family of distributions because each variable is governed by its own free parameter. The variational factors for preferences θ_{uk}, attributes β_{ik}, activity ξ_u, and popularity η_i are all Gamma distributions. (Thus their corresponding variational parameters are freely set scales and rates.) The variational factor for z_{ui} is a free multinomial, i.e., ϕ_{ui} is a K-vector that sums to one. This form stems from z_{ui} being a bank of Poisson variables conditional on a fixed sum y_{ui}, and the property that such conditional Poissons are distributed as a multinomial [17, 5].

After specifying the family, we fit the variational parameters $\nu = \{\lambda, \gamma, \kappa, \tau, \phi\}$ to minimize the KL divergence to the posterior, and then use the corresponding variational distribution $q(\cdot \mid \nu^*)$ as its proxy. The mean-field factorization facilitates both optimizing the variational objective and downstream computations with the approximate posterior, such as the recommendation score of Equation 1.

We optimize the variational parameters with a coordinate ascent algorithm, iteratively optimizing each parameter while holding the others fixed. The algorithm is illustrated in Figure 3. We denote shape with the superscript "shp" and rate with the superscript "rte". (We omit a detailed derivation due to space constraints.)

Note that our algorithm is very efficient on sparse matrices. In step 1, we need only update variational multinomials for the non-zero user/item observations y_{ui}. In steps 2 and 3, the sums over users and items need only to consider non-zero observations. This efficiency is thanks the likelihood of the full matrix only depending on the non-zero observations, as we discussed in the previous section.

We terminate the algorithm when the variational distribution converges. Convergence is measured by computing the prediction accuracy on a validation set. Specifically, we approximate the probability that a user consumed an item using the variational approximations to posterior expectations of θ_u and β_i, and compute the average predictive log likelihood of the validation ratings. The HPF algorithm stops when the change in log likelihood is less than 0.0001%. For the HPF and the BPF we find that the algorithm is largely insensitive to small changes in the hyper-parameters. To enforce sparsity, we set the shape hyperparameters a', a, c and c' to provide exponentially shaped prior Gamma distributions. We fixed each hyperparameter at 0.3. We set the hyperparameters b' and d' to 1, which fixes the prior mean at 1.

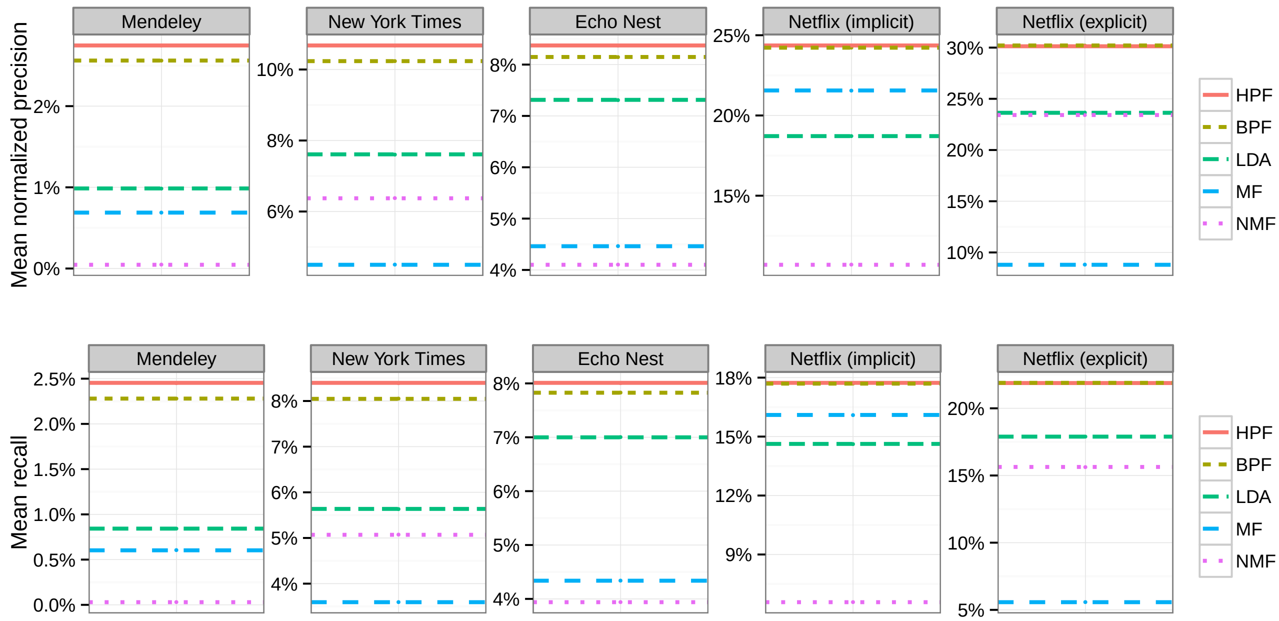

Figure 4: Predictive performance on data sets. The top and bottom plots show normalized mean precision and mean recall at 20 recommendations, respectively. While competing method performance varies across data sets, HPF and BPF consistently outperform competing methods.

4. EMPIRICAL STUDY

We evaluate the performance of the Hierarchical Poisson factorization (HPF) algorithm and its non-hierarchical variant (BPF) on a variety of large-scale user behavior data sets: users listening to music, users watching movies, users reading scientific articles, and users reading the newspaper. We find that HPF and BPF give significantly better recommendations than competing methods.

We first discuss the details of each data set and of the competing recommendation methods. We then describe our study, noting the superior performance and computational efficiency of HPF. We conclude with an exploratory analysis of preferences and attributes on several of the data sets.

Data Sets. We study the HPF algorithm in Figure 3 on several data sets of user behavior, including both implicit and explicit feedback:

- The **Mendeley** data set [16] of scientific articles is a binary matrix of 80,000 users and 260,000 articles, with 5 million observations. Each cell corresponds to the presence or absence of an article in a scientist's online library.
- The **Echo Nest** music data set [2] is a matrix of 1 million users and 385,000 songs, with 48 million observations. Each observation is the number of times a user played a song.
- The **New York Times** data set is a matrix of 1,615,675 users and 103,390 articles, with 80 million observations. Each observation is the number of times a user viewed an article.
- The **Netflix** data set [20] contains 480,000 users and 17,770 movies, with 100 million observations. Each observation is the rating (from 1 to 5 stars) that a user provided for a movie.

The scale and diversity of these data sets enables a robust evaluation of our algorithm. The Mendeley, Echo Nest, and New York Times data sets are sparse compared to the Netflix data. For example, we observe only 0.001% of all possible user-item ratings in Mendeley, while 1% of the ratings are non-zero in the Netflix data. This is partially a reflection of large number of items relative to number users in these data sets.

Furthermore, the intent signaled by an observed rating varies significantly across these data sets. For instance, the Netflix data set gives the most direct measure of stated preferences for items, as users provide an explicit star rating for movies they have watched. In contrast, article click counts in the New York Times data are a less clear measure of how much a user likes a given article—most articles are read only once, and a click through is only a weak indicator of whether the article was fully read, let alone liked. Ratings in the Echo Nest data presumably fall somewhere in between, as the number of times a user listens to a song likely reveals some indirect information about their preferences.

As such, we treat each data set as a source of implicit feedback, where an observed positive rating indicates that a user likes a particular item, but the rating value itself is ignored. The Mendeley data are already of this simple binary form. For the Echo Nest and New York Times data, we consider any song play or article click as a positive rating, regardless of the play or click count. We also consider two versions of the Netflix data—the original, explicit ratings, and an implicit version in which only 4 and 5 star ratings are retained as observations [26].

Competing methods. We compare Poisson factorization against an array of competing methods:

- **NMF**: Non-negative Matrix Factorization [21]. In NMF, user preferences and item attributes are modeled as non-negative vectors in a low-dimensional space. These latent vectors are randomly initialized and modified via an alternating multiplicative update rule to minimize the Kullback-Leibler divergence between the actual and modeled rating matrices.

- **LDA**: Latent Dirichlet Allocation [3]. LDA is a Bayesian probabilistic generative model where user preferences are represented by a distribution over different topics, and each topic is a distribution over items. Interest and topic distributions are randomly initialized and updated using stochastic variational inference [12] to approximate these intractable posteriors.

- **MF**: Probabilistic Matrix Factorization with user and item biases. We use a variant of matrix factorization popularized through the Netflix Prize [20], where a linear predictor—comprised of a constant term, user activity and item popularity biases, and a low-rank interaction term—is fit to minimize the mean squared error between the predicted and observed rating values, subject to L2 regularization to avoid overfitting. Weights are randomly initialized and updated via stochastic gradient descent using the Vowpal Wabbit package [33]. This corresponds to maximum a-posteriori inference under Probabilistic Matrix Factorization [30].

We note that while HPF, BPF, and LDA take only the non-zero observed ratings as input, traditional matrix factorization requires that we provide explicit zeros in the ratings matrix as negative examples for the implicit feedback setting. In practice, this amounts to either treating all missing ratings as zeros (as in NMF) and down-weighting to balance the relative importance of observed and missing ratings [14], or generating negatives by randomly sampling from missing ratings in the training set [7, 6, 26]. We take the latter approach for computational convenience, employing a popularity-based sampling scheme: we sample users by activity—the number of items rated in the training set—and items by popularity—the number of training ratings an item received to generate negative examples.[2]

Finally, we note a couple of candidate algorithms that failed to scale to our data sets. The fully Bayesian treatment of the Probabilistic Matrix Factorization [29], uses a MCMC algorithm for inference. The authors [29] report that a single Gibbs iteration on the Netflix data set with 60 latent factors, requires 30 minutes, and that they throw away the first 800 samples. This implies at least 16 days of training, while the HPF variational inference algorithm converges within 13 hours on the Netflix data. Another alternative, Bayesian Personalized Ranking (BPR) [27, 7], optimizes a ranking-based criteria using stochastic gradient descent. The algorithm performs an expensive bootstrap sampling step at each iteration to generate negative examples from the vast set of unobserved. We found time and space constraints to be prohibitive when attempting to use BPR with the data sets considered here.

[2]We also compared this to a uniform random sampling of negative examples, but found that the popularity-based sampling performed better.

Evaluation. Prior to training any models, we randomly select 20% of ratings in each data set to be used as a held-out test set comprised of items that the user has consumed. Additionally, we set aside 1% of the training ratings as a validation set and use it to determine algorithm convergence and to tune free parameters. We used the BPF and HPF settings described in Section 3.2 across all data sets.

During testing, we generate the top M recommendations for each user as those items with the highest predictive score under each method. For each user, we compute a variant of precision-at-M, which measures the fraction of relevant items in the user's top-M recommendations. So as not to artificially deflate this measurement for lightly active users who have consumed fewer than M items, we compute *normalized* precision-at-M, which adjusts the denominator to be at most the number of items that the user has in the test set. Likewise, we compute recall-at-M, which captures the fraction of items in the test set present in the top M recommendations.

Figure 4 shows the normalized mean precision at 20 recommendations for each method and data sets. We see that HPF and BPF outperform other methods on all data sets by a sizeable margin—as much as 8 percentage points. Poisson factorization provides high-quality recommendations—a relatively high fraction of items recommended by HPF are found to be relevant, and many relevant items are recommended. While not shown in these plots, the relative performance of methods within a data set is consistent as we vary the number of recommendations shown to users. We also note that while Poisson factorization dominates across all of these data sets, the relative quality of recommendations from competing methods varies substantially from one dataset to the next. For instance, LDA performs quite well on the Echo Nest data, but fails to beat classical matrix factorization for the implicit Netflix data set.

We also study precision and recall as a function of user activity to investigate how performance varies across users of different types. In particular, Figure 5 shows the mean normalized precision and mean recall at 20 recommendations as we look at performance for users of varying activity, measured by percentile. For example, the 10% mark shows mean performance across the bottom 10% of users, who are least active; the 90% mark shows the mean performance for all but the top 10% of most active users. Here we see that Poisson factorization outperforms other methods for users of all activity levels—both the "light" users who constitute the majority, and the relatively few "heavy" users who consume more—for all data sets.

For the New York Times and Netflix data, we see that higher levels of user activity enable us to better estimate user preferences and improve the quality of recommendations, as measured by mean normalized precision. For the Mendeley and Echo Nest data, however, we see a decline in normalized precision for the heavy users. One possible reason for this decline is that these data sets contain many more items than users, and this makes prediction difficult. For example, heavy users in the music data may have more diverse taste than the light users. We also note that the decrease in mean recall as we consider increasingly active sets of users is inevitable, as these individuals have a larger set of relevant items in the test set than their less active counterparts.

Exploratory analysis. The fitted model can be explored

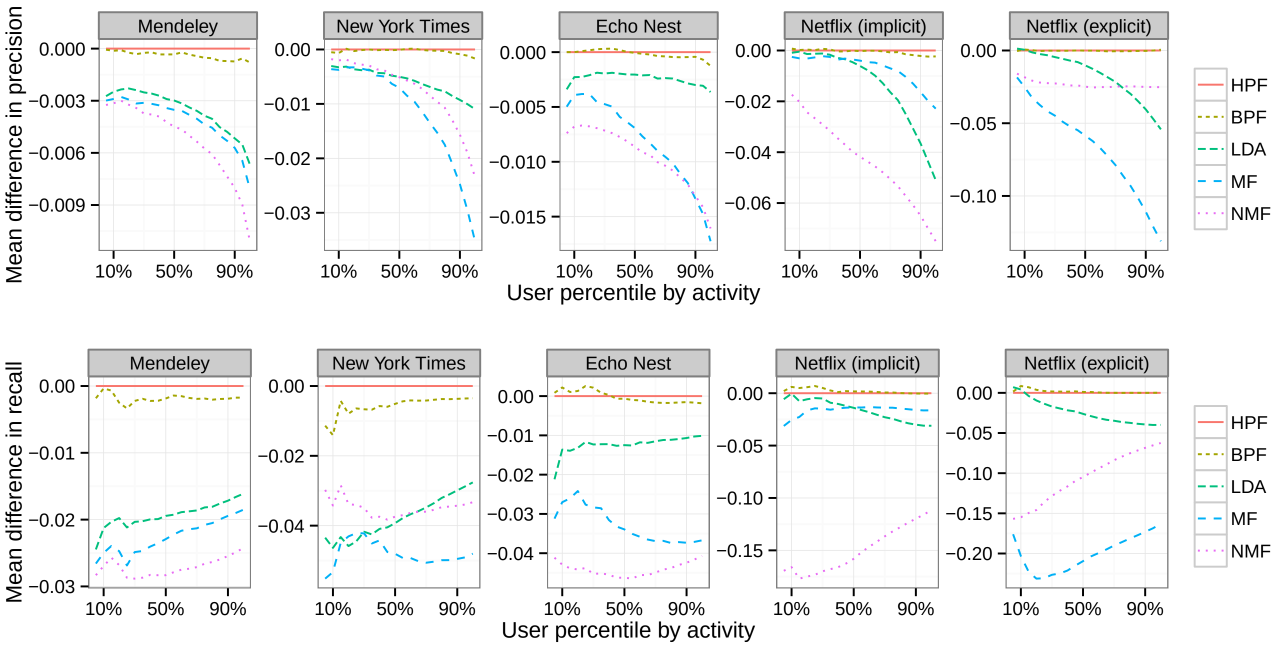

Figure 5: Predictive performance across users. The top and bottom plots show the mean difference in precision and recall to HPF at 20 recommendations, respectively, by user activity.

to discover latent structure among items and users and to confirm that the model is capturing the components in the data in a reasonable way. For example, in Figure 6 we illustrate the components discovered by our algorithm on the scientific articles in Mendeley and news articles in the New York Times. For each data set, the illustration shows the top items—items sorted in decreasing order of their expected weight β_i—from three of the 100 components discovered by our algorithm. From these, we see that learned components both cut across and differentiate between conventional topics and categories. For instance, in the New York Times data, we find that multiple business-related topics (e.g., self help and personal finance) comprise separate components, whereas other articles that appear across different sections of the newspaper (e.g., business and regional news) are unified by their content (e.g., airplanes).

5. CONCLUSION

We have demonstrated that Poisson factorization is an efficient and effective means of generating high quality recommendations across a variety of data sets ranging from movie views to scientific article libraries. It significantly outperforms existing recommendation methods on both explicit rating data and implicit behavior data, without the need for ad hoc modifications. Poisson factorization algorithms scale to massive data and differ from traditional methods in their ability to capture the heterogeneity amongst users and items, accounting for the wide range of activity and popularity amongst them, respectively.

Future work includes extensions to HPF, for example, to provide cold-start recommendations using text data [32], and to infer the number of latent components using Bayesian nonparametric assumptions [34].

6. REFERENCES

[1] B. Ball, B. Karrer, and M. Newman. Efficient and principled method for detecting communities in networks. *Physical Review E*, 84(3), Sept. 2011.

[2] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

[3] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.

[4] J. Canny. GaP: A factor model for discrete data. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2004.

[5] A. T. Cemgil. Bayesian inference for nonnegative matrix factorisation models. *Computational Intelligence and Neuroscience*, 2009, May 2009.

[6] G. Dror, N. Koenigstein, Y. Koren, and M. Weimer. The Yahoo! music dataset and KDD-cup '11. *Journal of Machine Learning Research*, 18:8–18, 2012.

[7] Z. Gantner, L. Drumond, C. Freudenthaler, and L. Schmidt-Thieme. Bayesian personalized ranking for non-uniformly sampled items. *JMLR W&CP*, Jan 2012.

[8] A. Gelman, X. Meng, and H. Stern. Posterior predictive assessment of model fitness via realized discrepancies. *Statistica Sinica*, 6:733–807, 1996.

[9] Z. Ghahramani and M. Beal. Propagation algorithms for variational Bayesian learning. In *Neural Information Processing Systems*, pages 507–513, 2001.

[10] P. K. Gopalan and D. M. Blei. Efficient discovery of overlapping communities in massive networks.

News articles from the *New York Times*	Scientific articles from Mendeley
"Business Self-Help"	**"Astronomy"**
Stay Focused And Your Career Will Manage Itself To Tear Down Walls You Have to Move Out of Your Office Self-Reliance Learned Early Maybe Management Isn't Your Style My Copyright Career	Theory of Star Formation Error estimation in astronomy: A guide Astronomy & Astrophysics Measurements of Omega from 42 High-Redshift Supernovae Stellar population synthesis at the resolution of 2003
"Personal Finance"	**"Biodiesel"**
In Hard Economy for All Ages Older Isn't Better It's Brutal Younger Generations Lag Parents in Wealth-Building Fast-Growing Brokerage Firm Often Tangles With Regulators The Five Stages of Retirement Planning Angst Signs That It's Time for a New Broker	Biodiesel from microalgae. Biodiesel from microalgae beats bioethanol Commercial applications of microalgae Second Generation Biofuels Hydrolysis of lignocellulosic materials for ethanol production
"All Things Airplane"	**"Political Science"**
Flying Solo Crew-Only 787 Flight Is Approved By FAA All Aboard Rescued After Plane Skids Into Water at Bali Airport Investigators Begin to Test Other Parts On the 787 American and US Airways May Announce a Merger This Week	Social Capital: Origins and Applications in Modern Sociology Increasing Returns, Path Dependence, and Politics Institutions, Institutional Change and Economic Performance Diplomacy and Domestic Politics Comparative Politics and the Comparative Method

Figure 6: The top 10 items by the expected weight β_i from three of the 100 components discovered by our algorithm for the New York Times and Mendeley data sets.

Proceedings of the National Academy of Sciences, 110(36):14534–14539, 2013.

[11] M. Hoffman. Poisson-uniform nonnegative matrix factorization. In *ICASSP*, 2012.

[12] M. Hoffman, D. Blei, and F. Bach. On-line learning for latent Dirichlet allocation. In *Neural Information Processing Systems*, 2010.

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

[14] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. *Data Mining*, Jan 2008.

[15] D. Inouye, P. Ravikumar, and I. Dhillon. Admixture of poisson mrfs: A topic model with word dependencies. In *Proceedings of The 31st International Conference on Machine Learning*, pages 683–691, 2014.

[16] J. Jack, Kris anwd Hammerton, D. Harvey, J. J. Hoyt, J. Reichelt, and V. Henning. MendeleyÊijs reply to the datatel challenge. *Procedia Computer Science*, 1(2):1–3, 2010.

[17] N. Johnson, A. Kemp, and S. Kotz. *Univariate Discrete Distributions*. John Wiley & Sons, 2005.

[18] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. Introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

[19] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434, New York, NY, USA, 2008. ACM.

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

[21] D. Lee and H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, October 1999.

[22] H. Ma, C. Liu, I. King, and M. R. Lyu. Probabilistic factor models for web site recommendation. In *Proceedings of the 34th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 265–274. ACM Press, 2011.

[23] J. Mairal, J. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010.

[24] B. Marlin, R. S. Zemel, S. Roweis, and M. Slaney. Collaborative filtering and the missing at random assumption. *arXiv preprint arXiv:1206.5267*, 2012.

[25] B. M. Marlin and R. S. Zemel. Collaborative prediction and ranking with non-random missing data. pages 5–12, 2009.

[26] U. Paquet and N. Koenigstein. One-class collaborative filtering with random graphs. . . . *of the 22nd international conference on* . . . , Jan 2013.

[27] S. Rendle, C. Freudenthaler, and Z. Gantner. . . . Bpr: Bayesian personalized ranking from implicit feedback. *Proceedings of the* . . . , Jan 2009.

[28] D. Rubin. Bayesianly justifiable and relevant frequency calculations for the applied statistician. *The Annals of Statistics*, 12(4):1151–1172, 1984.

[29] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proceedings of the 25th international conference on*

Machine learning, pages 880–887. ACM, 2008.

[30] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. *Advances in Neural Information Processing Systems*, 20:1257–1264, 2008.

[31] M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.

[32] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 448–456, New York, NY, USA, 2011. ACM.

[33] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1113–1120. ACM, 2009.

[34] M. Zhou, L. Hannah, D. B. Dunson, and L. Carin. Beta-negative binomial process and Poisson factor analysis. *Journal of Machine Learning Research - Proceedings Track*, 22:1462–1471, 2012.