HINRec: Scalable Recommendation in Heterogeneous Information Networks

Stephen Macke
University of Illinois at Urbana-Champaign
Urbana, IL 61801 USA
smacke@illinois.edu

ABSTRACT
We develop HINRec, a new recommendation model which is capable of incorporating extra relational information present in heterogeneous information networks (HINs) to improve recommendation quality. HINRec models sparse node behaviors in HINs, scaling with the total number of edges of all relations which participate in inference. HINRec explicitly models correlations in node behavior between different relations by allowing latent vector representations of the same node to participate in multiple relations. In contrast with previous models, HINRec captures multi-view user behavior and item perception in a completely probabilistic setting, as it allows nodes to express similar behavior between relations in different amounts. We demonstrate HINRec’s effectiveness at user-item recommendation on a variety of datasets with additional relational information present (e.g., user-friend or movie-director interaction). We give a scalable variational algorithm for posterior inference, and we show that it effectively leverages the additional data present in multi-relational HINs in order to improve prediction accuracy, outperforming previous Gamma-Poisson models and other strong baselines.

CCS Concepts
• Information systems → Collaborative filtering;

1. INTRODUCTION
Recommender systems have seen widespread deployment in a variety of content-based web services. They are vital for providing personalized experiences for users, allowing services to offer more items from the “long-tail” of different types of content, and further allowing users to effectively navigate this long-tail.

Recommendation systems based on collaborative filtering typically attempt to learn latent-vector representations of users and items by “embedding” users who interact with similar items close together in the same vector space, and similarly embedding items interacting with similar users close together.

One desirable property for recommendation systems is to be able to provide users with a brief reason for recommendations. For example, Amazon Prime Video (todo: maybe use different streaming service to avoid referring to Amazon twice) will recommend items similar to particular movies users watched in the past, and Amazon will have separate recommendations of items based on wishlist, browsing history, etc.

Briefly, a heterogeneous information network (HIN) is a graph with multiple node types and edge types, with each edge type corresponding to a relation between two (possibly non-identical) node types. Base recommendation models correspond to two node types (users and items) and a single edge type / relation (user-item interaction). A recommendation model which effectively integrates multiple relations from some HIN can provide more explicit reasons for suggested recommendations. For example, some users may prefer movies with a particular actor, and other users may prefer moves that their friends watch. By incorporating these extra relations (user-friend and movie-actor) in the same model, HINRec can learn user behaviors more explicitly and provide improved recommendations with reasons attached.

Our contributions in this paper are as follows:

1. We provide a fully generative process for explaining relational data from arbitrary heterogeneous network schemata which accounts for correlations in the relational data.

2. We give a scalable mean-field variational algorithm for posterior inference of latent vector mean values.

3. We show that HINRec learns high-quality posterior means for latent representations. The superior performance of HINRec on a variety of real-world datasets compared to existing methods demonstrates its effectiveness.

2. RELATED WORK
Generally speaking, recommendation facilitated via collaborative filtering falls into two major categories: memory-based approaches, and model-based approaches [13]. Memory-based approaches tend to feature relatively cheap offline computation at the cost of more expensive online computation, while model-based approaches tend to exhibit more expensive offline computation coupled with cheaper online
2.1 Collective Matrix Factorization

Collective matrix factorization was first described by Singh et al. in [14] in the context of relational learning. While vanilla matrix factorization can be seen as finding low-rank approximations of the data in a single relation $\mathcal{R}$, collective factorization finds low-rank approximations of several relations $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_n$ jointly. A given entity type (e.g., user or item) may participate in several relations concurrently depending on the relational schema, and each entity uses some nonempty subset (possibly all) of its latent representation in each relation in which it participates. HINRec may thus be seen as an instance of collective factorization, although the exact collective assumptions and optimization details differ.

In [19], the authors propose a Collective Bayesian Poisson Factorization (CBPF) for cold-start recommendation. Like HINRec, CBPF is a Gamma-Poisson generative model. Unlike HINRec, CBPF applies only to a single relational schema. Furthermore, CBPF does not exploit the full richness of potential collective assumptions, as latent vectors participating in multiple relations must use their entirety of their latent representations each time they participate. Lastly, CBPF requires manual tuning of hyperparameters which determine the weight of each relation in the final prediction given by the model, while HINRec learns these factors automatically on a per-entity basis.

2.2 Cross-Domain Collaborative Filtering

Prior work, such as [10, 7], focuses on leveraging overlapping user sets between multiple social domains to address sparsity challenges. While these techniques have been demonstrated to effectively transfer knowledge between “rating”-type relations, HINRec is more general and can transfer knowledge between arbitrary types of relations. HINRec is also effective at knowledge transfer between rating relations as demonstrated by our experiments on the Douban SNS dataset, and while perhaps not as affective as these previous methods, is more general in terms of the additional information it can incorporate.

2.3 Probabilistic Matrix Factorization

Probabilistic models for matrix factorization, such as [13, 12, 2, 18], typically assume Gaussian-distributed edge-weights or noises. HINRec, which attempts to capture sparse user-item interaction, is a Gamma-Poisson model, following in the footsteps of works such as [4, 6, 5, 19]. Unlike these works, HINRec generalizes to leverage any kind of side information from HINs with arbitrary schemata.

2.4 Recommendation in Heterogeneous Information Networks:

Other works in this area include [15, 17]. These methods, while able to incorporate arbitrary side information from HINs, suffer from some scalability drawbacks, as they will typically capture side information by examining metapaths. Longer metapaths tend to have a densifying affect on relation matrices – as the metapath-induced relations are not sparse, it is impossible to achieve the same level of scalability that Gamma-Poisson models achieve. HINRec, while capable of incorporating metapath-induced relations, does not require these, and can instead do inference on different types of unmodified sparse relations.

3. PRELIMINARIES

In this section, we define key terms and present background requisite for understanding this study.

3.1 Heterogeneous Information Network

We define information networks as follows:

**Definition 1. (Information Network).** An information network is defined as a directed graph $(G = (V, E))$ with an entity type mapping function $\phi : V \rightarrow A$ and a link type mapping function $\psi : E \rightarrow \mathcal{R}$. Each entity $v \in V$ belongs to an entity type $\phi(v) \in A$, and each link $l \in E$ belongs to a relation type $\psi(l) \in \mathcal{R}$.

We refer to information networks with $|A| > 1$ or $|\mathcal{R}| > 1$ as heterogeneous information networks (HINs). Traditional recommendation cast in the HIN framework typically has $|A| = 2$ and $|\mathcal{R}| = 1$; that is, traditional recommendation typically concerns bipartite graphs with a user type and an item type, between which there is a single type of interaction, such as a user rating an item.

3.2 Latent Vector Representations

In this paper, we are concerned with finding correlations in node behavior between different types of relations and exploiting these correlations to provide improved recommendation. In traditional model-based recommendation, the loose notion of “node linking behavior” (e.g., a user consuming an item induces a link in the network) is typically modeled by assuming latent vector representations of each node type (users and items). In a multi-typed HIN, however, we wish to capture different types of node linking behavior in the same latent representations. Toward this end, we will assume that latent vector representations of each node can be partitioned into different latent vector sections, each section capturing a particular view of that node’s linking behavior, which we define formally as follows:

**Definition 2. (Latent Vector Section)** A latent vector section of a node $i$’s latent vector representation $v^{(i)}$ is a contiguous subvector of components $\{v^{(i)}_j\}_{j \leq k}$, where $1 \leq j < k \leq \text{dim}(v^{(i)})$.

As we shall see later, one of the key ideas behind HINRec is to capture correlations in node linking behavior by allowing the same latent vector section to participate in different
latent representations, each latent representation capturing a node's linking behavior in a particular relation in the HIN. We call the set of relations in whose latent representations a particular latent vector section of a node participates a participating set.

**Definition 3. (Participating Set)** A participating set for a given relation contains the indices of a particular node type's latent vector sections which are used to generate edges for that relation. Because each relation requires a left node type and a right node type, we distinguish between left participating sets and right participating sets. For a given relation \( r \in \mathcal{R} \), we denote the left participating set by \( P_L(r) \) and the right participating set by \( P_R(r) \).

### 4. HINRec Model

In this section, we describe the HINRec model in the setting of a heterogeneous, multi-relational network. In short, for each relation \( r \in \mathcal{R} \) with nodes \( \{u\} \) of type \( a_{ur} \) and nodes \( \{v\} \) of type \( a_{vr} \), we represent each node \( u \) by a vector \( \Theta_u^{(r)} \) of \( K_r \) latent components, and similarly for each node \( v \). An edge weight is then drawn from a Poisson distribution with mean \( (\Theta_u^{(r)} \Theta_v^{(r)})^T \Theta_v^{(r)} \).

To facilitate recommendation in a heterogeneous setting, we further partition each \( \Theta_u \) into multiple latent sections \( \{\Theta_u^{(s)}\} \) where the \( s \)-th section is scaled by the corresponding scaling factor \( \sigma_u^{(r,s)} \). We allow a particular latent section \( \theta_{uk}^{(r,s)} \) to participate in a node's latent representation for multiple relations, not just \( r \). Each time a latent section participates in relation \( r \), we scale it by a distinct scaling factor \( \sigma_u^{(r,s)} \), allowing nodes to express different sorts of behavior in different amounts for each relation.

For example, in the Yelp dataset, customers typically interact with businesses in two different ways: either via tips, or by writing reviews. It is not unreasonable to assume that the tipping behavior of a user closely resembles that user’s review-writing behavior, and so we may impose the requirement that a user’s latent representation for tipping shares with that user’s latent representation for writing reviews. However, while tips are very short and composed from smartphones, writing a review requires more time and energy, so we expect a user to write fewer reviews than tips. As such, we want to allow the model to express similar user behavior in different amounts when interacting with these relations, so we allow it to scale the same latent section differently for each interaction.

#### 4.1 Generative Model

In this subsection we describe the statistical assumptions taken by HINRec used for generation of edges in a HIN. A summary of notation is given in table A. Our generative story is as follows:

1. For each node type \( a \in \mathcal{A} \):
   a. For each node \( u \) with \( \phi(u) = a \):
      i. For each vector section \( g_{uk}^{(a,s)} \in \{\theta_{uk}^{(a,n_u)} \} \):
         A. For each vector component \( g_{uk}^{(a,s)} \), draw
            \[ \theta_{uk}^{(a,s)} \sim \text{Gamma}(c^{(a,s)}, \phi^{(a,s)} / d^{(a,s)}) \]
      ii. For each section \( s \in P_L(r) \) draw scaling factor
          \[ \sigma_u^{(r,L,s)} \sim \text{Gamma}(c^{(a,s)}, \phi^{(a,s)} / d^{(a,s)}) \]
      iii. For each node \( v \) with \( \phi(v) = a_{vr} \):
         i. For each section \( s \in P_R(r) \) draw scaling factor
            \[ \sigma_v^{(r,R,s)} \sim \text{Gamma}(c^{(a,s)}, \phi^{(a,s)} / d^{(a,s)}) \]
   b. For each pair \( u, v \) with \( \phi(u) = a_{ul} \), \( \phi(v) = a_{vr} \):
      i. \( \Theta_u^{(r_L)} := \text{stack}(g_{uk}^{(a_L,n_u)}, \phi_{uk}^{(a_L,s)}) \) for \( s \in P_L(r) \)
      ii. \( \Theta_v^{(r_R)} := \text{stack}(g_{vk}^{(a_R,n_v)}, \phi_{vk}^{(a_R,s)}) \) for \( s \in P_R(r) \)
   c. Draw edge weight \( y_{uv}^{(r)} \sim \text{Pois}(\theta_u^{(r)} \Theta_v^{(r)}) \).

The above process describes the statistical assumptions about how the data are generated. The intuition is that nodes which exhibit similar behavior between multiple relations will share at least one latent vector section in their interactions between each of those relations. Nodes for which this correlation is more pronounced will draw larger corresponding scaling factors to more readily express this interaction, and nodes for which this correlation is less pronounced will draw smaller scaling factors in order to suppress this interaction.

#### 4.2 Modeling Correlated Behavior via Shared Vector Components

The key idea behind HINRec is to allow latent representations of nodes to share components when participating in different relations. The shared section of a node’s latent representation allows the model to capture commonalities in how a node expresses behavior in different settings. A user merely needs to specify the shared components as part of the modeling assumptions.

As an example, see figure 1. Each of the colored shapes indicates a vector component for a particular node type. Here, we have shared components between business-category and customer-business relations, indicating that the behavior of businesses with categories partially explains the behavior of businesses with customers. Similarly, the business-location relation shares components with the business-business relation, capturing the intuition that customers are more likely to interact with businesses in similar locations. A user merely needs to specify these shared components in order to capture assumptions about correlated behavior.

#### 4.3 Mean-Field Variational Methods for Inference

Recommender systems involve finding items for which the posterior expected dot products between latent representations are large. Requisite to performing this operation is a distribution over user and item latent vectors that explains the existing data. Unfortunately, however, computation of the exact posterior for HINRec is computationally intractable: the many-v-structures in the model create many large cliques when the data are observed, such that exact inference would require exhaustive search over a space exponential in the sizes of the cliques.

Variational inference is an optimization-based technique for posterior inference which restricts the search space of posterior distributions to a reasonable subset of the entire space, thus rendering the inference tractable. Mean-field
variational inference works by searching over distributions for which the corresponding graphical model is tree-shaped, and for which inference can therefore be performed efficiently [16]. Traditional treatments discuss mean-field variational inference as a method which minimizes the KL-divergence between a variational distribution (i.e., distribution in the tractable space over which search is performed) and the true posterior [8, 1]. More recent treatments discuss variational inference in a more optimization-based framework and make the distinction between mean-field techniques which search over posteriors whose graphical models are tree-structured, and naive mean-field techniques which search over posteriors whose graphical models are completely disconnected, corresponding to completely independent latent variables and priors [16].

For HINRec, we adopt the latter naive mean-field approach for computational convenience. Similar to [4], for each potential edge between nodes u and v in relation r, we introduce \( K(r) \) intermediate latent variables:

\[
z_{uvk}^{(r)} \sim \text{Pois} (\theta_{uk}^{(r)}(r) \theta_{vk}^{(r)})
\]

We further posit that edge weight \( y_{uv}^{(r)} = \sum_{k} z_{uvk}^{(r)} \). A sum of Poissons is Poisson with rate equal to the sum of rates of its constituents; thus, this introduction leaves the marginal distribution of the observations \( y_{uv}^{(r)} \) unchanged. Furthermore, with this introduction, the complete conditionals of all unobserved random variables in the model are in the exponential family of distributions, a property which permits posterior inference via a simple coordinate ascent algorithm [8].

We now describe the HINRec variational inference algorithm. A summary of notation is included in appendix A. Naive mean-field methods restrict our search over posterior distributions to those that factorize completely. That is, we posit a variational posterior distribution \( \pi \) over latent variables which factorizes:

\[
\pi(\theta, \sigma, z) = \pi(\theta)\pi(\sigma)\pi(z)
\]

\[
\pi(\theta) = \prod_{a \in A} \prod_{s=1}^{u_a} \prod_{u \in S_a} \pi(\theta_u^{(a,s)})
\]

\[
\pi(\sigma) = \prod_{r \in R} \prod_{s=1}^{P_r^{(r)}} \pi(\sigma^{(r,L,s)}) \prod_{s=1}^{P_R^{(r)}} \pi(\sigma^{(r,R,s)})
\]

\[
\pi(z) = \prod_{u \in S_{ar_L}} \prod_{v \in S_{ar_R}} \pi(z_{uv}^{(r)})
\]

Each variable in the variational distribution is governed by its own set of variational parameters. To declutter the notation, we denote the variational parameters for a given variable with the same symbol as that variable, but with an overhead “hat”. For example, the variational distribution of a given \( \theta_u^{(a,s)} \) is gamma, and we parameterize it with \( \hat{a}, \hat{s} \quad \text{gamma distribution.} \) The variables \( \hat{\sigma}_{L,s}^{(r,L,s)} \) and \( \hat{\sigma}_{R,s}^{(r,R,s)} \) are similarly parameterized.

We now describe the HINRec variational inference algorithm. A summary of notation is included in appendix A. Naive mean-field methods restrict our search over posterior distributions to those that factorize completely. That is, we posit a variational posterior distribution \( \pi \) over latent variables which factorizes:

\[
\pi(\theta, \sigma, z) = \pi(\theta)\pi(\sigma)\pi(z)
\]

\[
\pi(\theta) = \prod_{a \in A} \prod_{s=1}^{u_a} \prod_{u \in S_a} \pi(\theta_u^{(a,s)})
\]

\[
\pi(\sigma) = \prod_{r \in R} \prod_{s=1}^{P_r^{(r)}} \pi(\sigma^{(r,L,s)}) \prod_{s=1}^{P_R^{(r)}} \pi(\sigma^{(r,R,s)})
\]

\[
\pi(z) = \prod_{u \in S_{ar_L}} \prod_{v \in S_{ar_R}} \pi(z_{uv}^{(r)})
\]

Each variable in the variational distribution is governed by its own set of variational parameters. To declutter the notation, we denote the variational parameters for a given variable with the same symbol as that variable, but with an overhead “hat”. For example, the variational distribution of a given \( \theta_u^{(a,s)} \) is gamma, and we parameterize it with \( \hat{a}, \hat{s} \quad \text{gamma distribution.} \) The variables \( \hat{\sigma}_{L,s}^{(r,L,s)} \) and \( \hat{\sigma}_{R,s}^{(r,R,s)} \) are similarly parameterized.

We now describe the HINRec variational inference algorithm. A summary of notation is included in appendix A. Naive mean-field methods restrict our search over posterior distributions to those that factorize completely. That is, we posit a variational posterior distribution \( \pi \) over latent variables which factorizes:

\[
\pi(\theta, \sigma, z) = \pi(\theta)\pi(\sigma)\pi(z)
\]

\[
\pi(\theta) = \prod_{a \in A} \prod_{s=1}^{u_a} \prod_{u \in S_a} \pi(\theta_u^{(a,s)})
\]

\[
\pi(\sigma) = \prod_{r \in R} \prod_{s=1}^{P_r^{(r)}} \pi(\sigma^{(r,L,s)}) \prod_{s=1}^{P_R^{(r)}} \pi(\sigma^{(r,R,s)})
\]

\[
\pi(z) = \prod_{u \in S_{ar_L}} \prod_{v \in S_{ar_R}} \pi(z_{uv}^{(r)})
\]

Each variable in the variational distribution is governed by its own set of variational parameters. To declutter the notation, we denote the variational parameters for a given variable with the same symbol as that variable, but with an overhead “hat”. For example, the variational distribution of a given \( \theta_u^{(a,s)} \) is gamma, and we parameterize it with \( \hat{a}, \hat{s} \quad \text{gamma distribution.} \) The variables \( \hat{\sigma}_{L,s}^{(r,L,s)} \) and \( \hat{\sigma}_{R,s}^{(r,R,s)} \) are similarly parameterized.

We now describe the HINRec variational inference algorithm. A summary of notation is included in appendix A. Naive mean-field methods restrict our search over posterior distributions to those that factorize completely. That is, we posit a variational posterior distribution \( \pi \) over latent variables which factorizes:

\[
\pi(\theta, \sigma, z) = \pi(\theta)\pi(\sigma)\pi(z)
\]

\[
\pi(\theta) = \prod_{a \in A} \prod_{s=1}^{u_a} \prod_{u \in S_a} \pi(\theta_u^{(a,s)})
\]

\[
\pi(\sigma) = \prod_{r \in R} \prod_{s=1}^{P_r^{(r)}} \pi(\sigma^{(r,L,s)}) \prod_{s=1}^{P_R^{(r)}} \pi(\sigma^{(r,R,s)})
\]

\[
\pi(z) = \prod_{u \in S_{ar_L}} \prod_{v \in S_{ar_R}} \pi(z_{uv}^{(r)})
\]

Each variable in the variational distribution is governed by its own set of variational parameters. To declutter the notation, we denote the variational parameters for a given variable with the same symbol as that variable, but with an overhead “hat”. For example, the variational distribution of a given \( \theta_u^{(a,s)} \) is gamma, and we parameterize it with \( \hat{a}, \hat{s} \quad \text{gamma distribution.} \) The variables \( \hat{\sigma}_{L,s}^{(r,L,s)} \) and \( \hat{\sigma}_{R,s}^{(r,R,s)} \) are similarly parameterized.

We now describe the HINRec variational inference algorithm. A summary of notation is included in appendix A. Naive mean-field methods restrict our search over posterior distributions to those that factorize completely. That is, we posit a variational posterior distribution \( \pi \) over latent variables which factorizes:

\[
\pi(\theta, \sigma, z) = \pi(\theta)\pi(\sigma)\pi(z)
\]

\[
\pi(\theta) = \prod_{a \in A} \prod_{s=1}^{u_a} \prod_{u \in S_a} \pi(\theta_u^{(a,s)})
\]

\[
\pi(\sigma) = \prod_{r \in R} \prod_{s=1}^{P_r^{(r)}} \pi(\sigma^{(r,L,s)}) \prod_{s=1}^{P_R^{(r)}} \pi(\sigma^{(r,R,s)})
\]

\[
\pi(z) = \prod_{u \in S_{ar_L}} \prod_{v \in S_{ar_R}} \pi(z_{uv}^{(r)})
\]

Each variable in the variational distribution is governed by its own set of variational parameters. To declutter the notation, we denote the variational parameters for a given variable with the same symbol as that variable, but with an overhead “hat”. For example, the variational distribution of a given \( \theta_u^{(a,s)} \) is gamma, and we parameterize it with \( \hat{a}, \hat{s} \quad \text{gamma distribution.} \) The variables \( \hat{\sigma}_{L,s}^{(r,L,s)} \) and \( \hat{\sigma}_{R,s}^{(r,R,s)} \) are similarly parameterized.

We now describe the HINRec variational inference algorithm. A summary of notation is included in appendix A. Naive mean-field methods restrict our search over posterior distributions to those that factorize completely. That is, we posit a variational posterior distribution \( \pi \) over latent variables which factorizes:

\[
\pi(\theta, \sigma, z) = \pi(\theta)\pi(\sigma)\pi(z)
\]

\[
\pi(\theta) = \prod_{a \in A} \prod_{s=1}^{u_a} \prod_{u \in S_a} \pi(\theta_u^{(a,s)})
\]

\[
\pi(\sigma) = \prod_{r \in R} \prod_{s=1}^{P_r^{(r)}} \pi(\sigma^{(r,L,s)}) \prod_{s=1}^{P_R^{(r)}} \pi(\sigma^{(r,R,s)})
\]

\[
\pi(z) = \prod_{u \in S_{ar_L}} \prod_{v \in S_{ar_R}} \pi(z_{uv}^{(r)})
\]
The coordinate ascent algorithm for HINRec posterior inference is given in algorithm 1. It works iteratively cycling through each variable’s variational parameters, updating these parameters while holding all the others fixed. The coordinate ascent algorithm is guaranteed to improve a lower bound on the marginal likelihood of the variational distribution, which in turn implies decreasing the KL divergence between the variational distribution and true posterior.

The algorithm is run until convergence. Convergence is measured by change in predictive likelihood on a sample of the validation set.

Once the algorithm converges, recommendation is facilitated by finding the mean values of each Θ under the learned variational distribution, and recommending items whose learned latent representations have large inner product. That is, for user u, we recommend items i for which the following is large:

\[
\sum_k \mathbb{E}_\pi[\Theta_{uk}\Theta_{ik}] = \sum_k \mathbb{E}_\pi[\Theta_{uk}][\mathbb{E}_\pi[\Theta_{ik}]]
\]

### 4.4 Implementation Details

Excepting bookkeeping for subscripts and superscripts, most of algorithm 1 is straightforward. Still, there are a couple of details worth discussing in more detail.

**Update for Rate Parameters of Scaling Factors:**

The update for the rate variational parameter of each scaling factor is given as

\[
\mathbf{rt}_u \sigma_u^{(r,s)} \leftarrow \mathbf{rt}_u \sigma_u^{(r,s)} + \sum_k \frac{\mathbf{rb}_v^{(r,s)} \mathbf{sh}_v}{\mathbf{rt}_v \sigma_v^{(r,s)} \mathbf{rt}_v \theta_{vk}}
\]

which, if implemented naively, requires repeated computation of the sum \( \sum_k \frac{\mathbf{rb}_v^{(r,s)} \mathbf{sh}_v}{\mathbf{rt}_v \sigma_v^{(r,s)} \mathbf{rt}_v \theta_{vk}} \) once for each node u. For relations which do not self-loop in the HIN schema (that is, relations between distinct node types), this sum does not change for each node u and can be cached. For relations which do self-loop, however, this sum is over all nodes \( v \neq u \), and changes for each node u. In our implementation, we avoid repeated computation of this sum for each u as follows:

- Compute \( S \leftarrow \sum_v \sum_k \frac{\mathbf{rb}_v^{(r,s)} \mathbf{sh}_v}{\mathbf{rt}_v \sigma_v^{(r,s)} \mathbf{rt}_v \theta_{vk}} \) once (including for \( v = u \))

- For each node u:

  1. \( S \leftarrow S - \sum_k \frac{\mathbf{rb}_v^{(r,s)} \mathbf{sh}_v}{\mathbf{rt}_v \sigma_v^{(r,s)} \mathbf{rt}_v \theta_{vk}} \)
  2. \( \mathbf{rt}_u \sigma_u^{(r,s)} \leftarrow \mathbf{rt}_u \sigma_u^{(r,s)} + S \)
  3. \( S \leftarrow S + \sum_k \frac{\mathbf{rb}_v^{(r,s)} \mathbf{sh}_v}{\mathbf{rt}_v \sigma_v^{(r,s)} \mathbf{rt}_v \theta_{vk}} \)

This procedure preserves correctness of inference while using a simple computational trick to drastically decrease computational complexity.

**Parallelization:** One advantage of the simple updates of naive mean-field algorithms is that they are typically easy to parallelize. Algorithm 1 is no exception. In particular, computation of the \( \mathbf{z}_u^{(r)} \) is embarrassingly parallel over all the edges in the HIN. In our implementation, we perform this computation on the GPU using CUDA, which cuts time for each iteration of inference roughly in half on our large dataset (Douban), from 30 seconds to roughly 15 seconds.

### 5. EMPIRICAL STUDY

#### 5.1 Datasets

We evaluate HINRec on the Yahoo! Movies Webscope dataset (heretofore referred to as Yahoo!), user and book ratings derived from the Douban social networking service (referred to as Douban), and the 2015 Yelp Academic Challenge dataset (referred to as Yelp). The scale and diversity of the datasets demonstrates HINRec’s effectiveness at incorporating a variety of different types of relational information for improved predictive performance on a variety of
sparse user-item recommendation tasks. We briefly summarize each dataset relation type used in HINRec below. Schemata are given in appendix B. We use standard 80-10-10 train/dev/test splits (sampled randomly).

5.1.1 Yahoo!
The Yahoo! dataset is the smallest, with 7,642 users and 11,916 movies. Additionally, each movie is augmented with some additional attributes such as actors and directors from which we can derive additional relations to include in the HINRec framework. We include the movie-director relation when training HINRec, which includes a total of 5,249 directors. We attempt to predict top movies for each user that do not appear in the train set.

5.1.2 Douban
The Douban SNS dataset contains users, movies, books, and songs, and ratings between users and each other node type. We train HINRec on user-movie and user-book ratings, which includes a total of 39,771 users, 56,489 movies, 166,841 books, and 5,212,999 ratings. We attempt to predict the top-rated movies for each user that do not appear in the train set.

5.1.3 Yelp
The Yelp Academic dataset includes information on reviews, tips, and friends. In our experiments, we consider review and friend relations from 269,231 users and 21,799 restaurants with 792,745 reviews and 270,490 friendship links. We discard ratings and try to predict restaurants for which a user will write reviews. To set friendship weights, we use Jaccard similarity between the set restaurants a user has written reviews for with the set the user’s friend wrote reviews for (for reviews that appear in the train set).

5.2 Baselines
We compare HINRec to Hierarchical Poisson Factorization (HPF) which has similar scaling properties but cannot incorporate arbitrary relational information, and the non-negative matrix factorization (NMF) model implemented in SKLearn [11, 3], which uses a coordinate descent solver to minimize the RMSE between the original matrix and that reconstructed from user-item factored representations.

5.3 Results
The results of our experiments are summarized in table 1. HINRec’s superior performance indicates it’s ability to learn effective recommendation strategy from arbitrary types of side information.

6. CONCLUSION AND FUTURE WORK
We presented HINRec, a new recommendation model for modeling sparse user-item interaction which scales to massive datasets and is capable of incorporating arbitrary side-information. We demonstrated HINRec’s effectiveness at incorporating extra information on several datasets of varying scale and HIN schemata.

There are several directions for extending and strengthening this work. One is to provide some guidance for hyperparameter tuning (in particular, how to set dimensions of latent dimensions). Another is to extend HINRec from a matrix-factorization approach to a tensor-factorization method which accounts for changing user tastes through time. We should also visualize the learned latent representations with some embedding method such as t-SNE to validate the assumption that similar node types participating in non-target relations (e.g., directors in our Yahoo! Movies dataset) will be close to each other in the latent space, and that it makes sense to learn joint latent representations as opposed to incorporating simpler categorical features. Lastly, we should evaluate against more baselines, particularly some which also attempt to take into account side information in HINs.

7. REFERENCES
APPENDIX

A. SUMMARY OF NOTATION

A summary of notation used in the HINRec generative model as well as the inference procedure is given in table 2.

B. SCHEMATA OF HINS USED IN EMPIRICAL STUDY

We diagram of schemata of our HIN datasets in figure 2, noting the sub-chemata actually used for learning in our experiments.

C. COMPARISON WITH HIERARCHICAL POISSON FACTORIZATION

In the context of the vanilla Poisson factorization model given in [4], the scaling factors for the generalized model replace the roles of user activities or item popularities. To see this, consider the simple case of a single relation between a user type and an item type, where each nodes of each type generate links with nodes of the other type using latent vectors consisting of a single latent section and scaling factor. In this case, note that we could simplify the generative process as follows:

1. For each user $u$:
   
   (a) Sample activity / scaling factor $\sigma_u \sim \Gamma(a', c'/d')$.
   
   (b) For each vector component $k$, sample preference $\Theta_{uk} \sim \Gamma(a', a'/b' \sigma_u) \sim \sigma_u \Gamma(a', a'/b')$

2. For each item $i$:
   
   (a) Sample popularity / scaling factor $\sigma_i \sim \Gamma(c', c'/d')$.
   
   (b) For each vector component $k$, sample attribute $\Theta_{ik} \sim \Gamma(a', a'/b' \sigma_i) \sim \sigma_i \Gamma(a', a'/b')$

3. For each user $u$ and item $i$, sample rating / link $y_{ui} \sim \text{Pois} (\Theta_u^T \Theta_i)$

This uses the elementary fact that scaling a Gamma random variable by factor $\sigma$ is equivalent to dividing its rate parameter by $\sigma$ before sampling. Thus, when considering the simple case, while vanilla Poisson factorization effectively scales up the rate parameter of the Gamma from which components are sampled by user/item activity/popularity, HINRec scales it down by this amount. That is, vanilla Poisson factorization would sample user vector components from a simple case, while vanilla Poisson factorization effectively scales up the rate parameter of the Gamma from which components are sampled by user/item activity/popularity, HINRec scales it down by this amount. That is, vanilla Poisson factorization would sample user vector components from

$$\Gamma(a', a'/b' \sigma_u) \sim \sigma_u \Gamma(a', a'/b').$$
Symbol | Semantics
---|---
\(A\) | set of node types
\(\mathcal{R}\) | set of relations
\(\phi\) | node typing function mapping nodes to types \(a \in A\)
\(\psi\) | edge typing function mapping edges to relations \(r \in \mathcal{R}\)
\(S_u\) | set of nodes \(u\) with \(\phi(u) = a \in A\)
\(\alpha_{L}\) | node type of left side of relation \(r\)
\(\alpha_{R}\) | node type of right side of relation \(r\)
\(\sigma_{u}^{(L,s)}\) | scaling factor for latent section \(s\) of node \(u\) used in left side of relation \(r\)
\(\sigma_{u}^{(R,s)}\) | scaling factor for latent section \(s\) of node \(u\) used in right side of relation \(r\)
\(\Theta_{u}^{(L)}\) | vector of stacked, scaled latent vector sections for node \(u\) which participates in the left side of relation \(r\)
\(\Theta_{u}^{(R)}\) | vector of stacked, scaled latent vector sections for node \(u\) which participates in the left right of relation \(r\)
\(y_{uv}\) | edge weight between nodes \(u\) and \(v\) in relation \(r\)
\(z_{(r,s)}^{uv}\) | vector of multinomial variational parameters for section \(s\) latent contributions to edge weights
\(\hat{\theta}_{u}^{(a,s)}\) | vector of \(\{\text{shape, rate}\}\) variational parameters for latent section \(s\) of node \(u\)
\(\hat{\sigma}_{u}^{(r,s)}\) | node \(u\)’s section \(s\) \(\{\text{shape, rate}\}\) scaling variational parameter
\(\hat{\Theta}_{u}^{(a,s)}\) | stacked vector of \(\hat{\theta}_{u}^{(a,s)}\) for each relevant section \(s\)
\(\hat{\sigma}_{u}^{(r,s)}\) | stacked vector of \(\hat{\sigma}_{u}^{(r,s)}\) for each relevant section \(s\)
\(\Psi\) | digamma function
\(\pi\) | distribution over variational parameters

Table 2: Summary of notation used in HINRec generative model

Figure 2: Datasets used to evaluate HINRec. The portions of the schemata from which we take relations used to train HINRec are shown in the boxed section of each schema.