Bipartite Sparse Exchangeable Graphs for Machine Learning

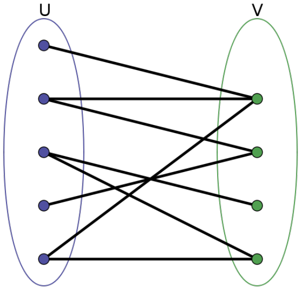
Ekansh Sharma
with Victor Veitch, Zacharie Naulet, and Daniel Roy

ML Group Meeting

Bipartite graphs

Definition

A bipartite graph is an ordered triple (V_U, V_I, E) where V_U, V_I are sets of vertices and $E \subseteq V_U \times V_I$ is a set of edges.



- $u \in V_U$ means that there is a vertex with label u in the graph.
- $t \in V_I$ means that there is a vertex with label t in the graph.
- $(u, t) \in E$ means that there is an edge between vertices u and t.
- Note that there are edges only between vertices of different part.
- I'll use the vocab of *users* and *items* to distinguish between parts of a graph.

Overview

- Many models in machine learning, e.g. models for topic modeling, probabilistic matrix factorization, feature allocation, define probability distribution over bipartite graphs

- Interested in inferring the parameters of distribution over bipartite graph from direct or indirect observation of the graph

Overview

- Many models in machine learning, e.g. models for topic modeling, probabilistic matrix factorization, feature allocation, define probability distribution over bipartite graphs
- Interested in inferring the parameters of distribution over bipartite graph from direct or indirect observation of the graph
- Recent work in statistical network modeling has introduced the sparse exchangeable (graphex) models as a new family of probability distributions over graphs (in particular, over bipartite graphs)

Big Picture

- What practical insights can be gained in light of the new sparse graph theory?
  - How to check for sparsity for a fixed size graph?
  - Can we scale inference for models belonging to this new model class?

Background

Example: Non-Negative Matrix Factorization

- Observed Data: Binary Adjacency matrix indicating if user i consumed item j
- Model: Probability of an edge depends only on the inner product of some latent user preference and item property
- Task: Make customized recommendations of items to users based on the observed graph

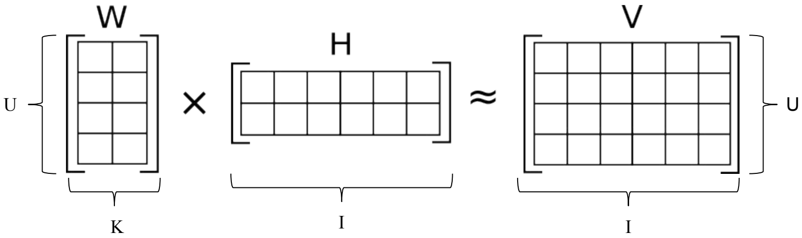


Figure: Non-Negative Matrix Factorization

Example: Non-Negative Matrix Factorization

Generative Model

For an observed adjacency matrix with n users and m items,

- Each user $i \in \{1, \ldots, n\} = V_U$
  User preference: $X_i \in \mathbb{R}_+^K$, is assigned iid with distribution $\Pr(X_i \in \cdot | \theta_U)$
- Each item $j \in \{1, \ldots, m\} = V_I$
  Item property: $Y_j \in \mathbb{R}_+^K$, is assigned iid with distribution $\Pr(Y_j \in \cdot | \theta_I)$
- Given the user preference and item property, include the edge (i, j) in E with probability $W(X_i^T Y_j)$, for e.g. $W(X_i^T Y_j) = 1 - e^{-X_i^T Y_j}$

Task: Learn likely values of $\{X_i\}, \{Y_j\}, \theta_U$, and θ_I

Dense Exchangeable Bipartite Graph Framework

Generative Model

Sample of a random graph $G_{n,m}$, where $n, m \in \mathbb{N}$, as follows:

- For each user vertex $i \in \{1, \ldots, n\}$, assign a latent feature $x_i \in \mathcal{X}$ iid with some distribution
- For each item vertex $j \in \{1, \ldots, m\}$, assign a latent feature $y_j \in \mathcal{Y}$ iid with some distribution
- Given the user feature and item feature, include the edge (i, j) independently with probability $W(x_i, y_j)$, where $W : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ is called a graphon

Notes:

- Examples include, probabilistic matrix factorization, topic modeling, feature allocation
- Gives rise to graphs that are almost surely dense graphs, i.e., as the number of vertices increase, the ratio $\rho = \frac{|E|}{|V_U||V_I|}$ remains constant.

Sparse Exchangeable Bipartite Graph Framework

Generative Model

Sample a random graph $G_{s,\alpha}$, where $s, \alpha \in \mathbb{R}_+$, as follows:

- Sample user features: a Poisson Process $\{(x_i, u_i)\}$ on $\mathcal{X} \times [0, s]$
- Sample item features: a Poisson Process $\{(y_j, t_j)\}$ on $\mathcal{Y} \times [0, \alpha]$
- Given the user feature and item feature, include the edge (u_i, t_j) independently with probability $W(x_i, y_j)$, where $W : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ is a (generalized) graphon
- Throw away all vertices that failed to connect

Example

$W : \mathbb{R}_+ \times \mathbb{R}_+ \to [0, 1]$ is defined as: $W(x, y) = 1 - \exp(-xy)$
User Size: $s < s' \in \mathbb{R}_+$
Item Size: $\alpha < \alpha' \in \mathbb{R}_+$

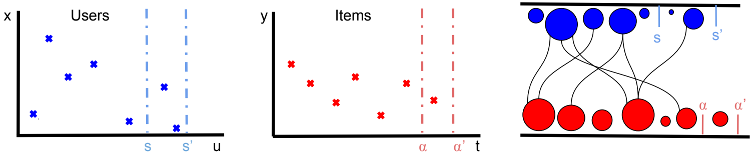


Figure: Sample from the generative model

Example

$$V_U^s = \{u_2, u_3, u_4\} \quad V_U^{s'} = \{u_2, u_3, u_4, u_5\}$$
$$V_I^\alpha = V_I^{\alpha'} = \{t_1, t_2, t_4, t_6\}$$
$$E^s = \{(u_2, t_1), (u_2, t_4), (u_2, t_6), (u_3, t_1), (u_4, t_2), (u_4, t_4)\}$$
$$E^{s'} = \{(u_2, t_1), (u_2, t_4), (u_2, t_6), (u_3, t_1), (u_4, t_2), (u_4, t_4), (u_5, t_4)\}$$

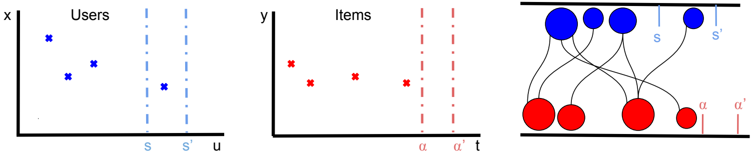


Figure: Sample from the generative model

Sampling Results

Questions

Sparsity is a property of sequences of graphs.

- How can we test if a given dataset is generated using a sparse model?
- How to sub-sample the dataset for test-train split?

p-Sampling a Graph

Definition

A user p-sampling of a graph G is a random subgraph given by selecting each user vertex of G independently with probability p, and then returning the induced edge set

p-Sampling a Graph

Definition

A user p-sampling of a graph G is a random subgraph given by selecting each user vertex of G independently with probability p, and then returning the induced edge set

Theorem (VR16)

Let $(G_{s,\alpha})_{s\in\mathbb{R}_+,\alpha\in\mathbb{R}_+}$ *be generated by* W*. If* $\mathrm{user\text{-}p\text{-}samp}(G_{s,\alpha}, r/s)$ *is an* r/s*-sampling of* G_s*, then*

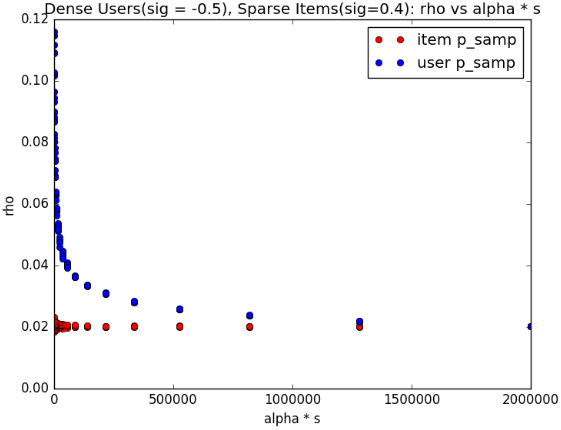
$$\mathrm{user\text{-}p\text{-}samp}(G_{s,\alpha}, r/s) \stackrel{d}{=} G_{r,\alpha}$$

Mutatis mutandis for $\mathrm{item\text{-}p\text{-}samp}$

Test for Sparsity

Recall: For densely generated graphs, $\rho = \frac{|E|}{|V_U| * |V_I|}$ remains constant.

For sparsely generated graphs, as user/item size decreases, ρ should increase.



Test for Sparsity

Do we actually observe datasets that were sparsely generated?

Test for Sparsity

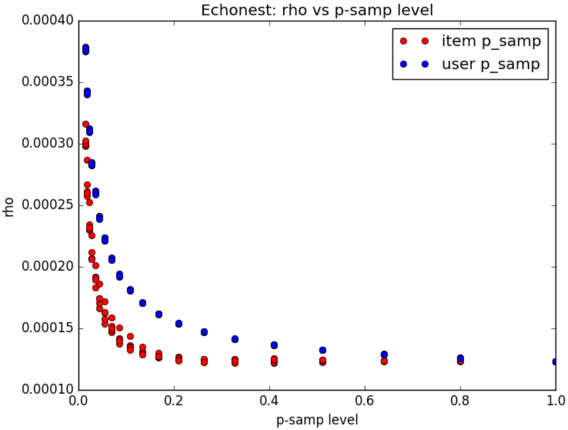
Do we actually observe datasets that were sparsely generated?

Yes!

Test for Sparsity

Do we actually observe datasets that were sparsely generated?

Yes! : Echonest: Music data set with 1 million users and 385,000 songs, with 48 million observations. Each observation is the number of times a user played a song.



Test for Sparsity

Netflix: Movie data set with 480,000 users and 17,770 movies, with 100 million observations.

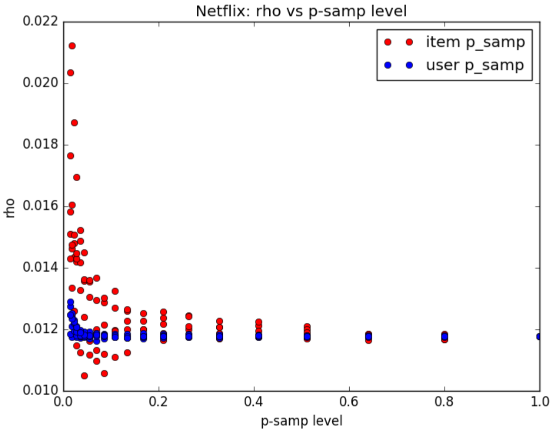


Figure: Netflix

Insights related to Train-Test Split

How do we split graph datasets like Echonest, Netflix?

- Dataset is the edge set $E = \{(i, j)\}_{i \in V_U, j \in V_I}$

Insights related to Train-Test Split

How do we split graph datasets like Echonest, Netflix?

- Dataset is the edge set $E = \{(i, j)\}_{i \in V_U, j \in V_I}$
- Existing Approach: Randomly holdout some fraction of the edges
  - Makes the recommendation task easier: more likely to select edges from high degree users and high degree items.
  - Unfairly penalizes generative models: Posterior distribution would be incorrect because of biased sampling

Insights related to Train-Test Split

How do we split graph datasets like Echonest, Netflix?

- Dataset is the edge set $E = \{(i,j)\}_{i \in V_U, j \in V_I}$
- Existing Approach: Randomly holdout some fraction of the edges
  - Makes the recommendation task easier: more likely to select edges from high degree users and high degree items.
  - Unfairly penalizes generative models: Posterior distribution would be incorrect because of biased sampling
- Proposed Approach: Based on $p-$sampling
  - Split the edge set into Train and Test set by p-sampling users
  - Run the training procedure to infer item properties. Fix the item properties once training finishes.
  - Split Test set into lookup and heldout set by p-sampling items
  - Run the training procedure to infer users preferences in the lookup set (keeping item properties fixed)
  - Perform the prediction task on heldout set.

Extended Example: Sparse Non-Negative Matrix Factorization

Application: Sparse Non-Negative Matrix Factorization

Recall:

- Observed Data: Adjacency matrix g where g_{ij} indicates user i consumed item j

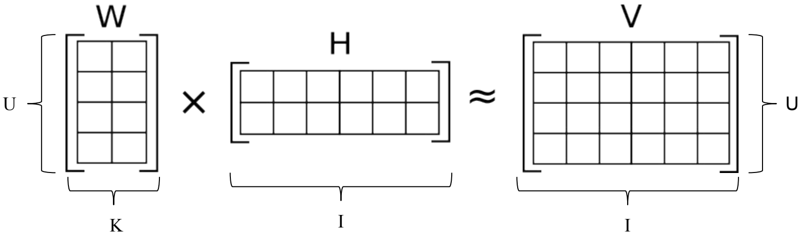


Figure: Non-Negative Matrix Factorization

Application: Non-Negative Matrix Factorization

Main Idea

Users and items have a $K-$dimensional latent feature

- Each user i has total popularity P_i and feature affinity $(\phi_{ik})_{k=1}^K$
- Each item j has total popularity P'_j and feature affinity $(\varphi_{jk})_{k=1}^K$
- Probability of edge (i, j) is given as $1 - exp(-P_i P'_j \sum_k \phi_{ik}\varphi_{jk})$

Note: Inclusion probability of edge (i, j) is the probability of a non-zero draw from a Poisson distribution with mean $P_i P'_j \sum_k \phi_{ik}\varphi_{jk}$

Digression: Generalized Gamma Process

For $(\sigma, \tau) \in (0,1) \times \mathbb{R}_+ \cup [\infty \times \mathbb{R}_+$, let $\rho_{\sigma,\tau}$ denote the Lévy intensity measure of a Generalized Gamma process:

$$\rho_{\sigma,\tau}(\mathrm{d}\omega) = \frac{1}{\Gamma(1-\sigma)} \omega^{-1-\sigma} \exp(-\omega\tau)\mathrm{d}\omega,$$

Properties of GGP

- Pseudo-conjugacy relationship with Poisson distribution - useful for inference
- Poisson process has finite activation $\iff$ $\sigma < 0$ (model is equivalent to dense graph) - useful for interpretablity

Application: Sparse Non-Negative Matrix Factorization

Generative Model (TC16)

- Sample user features:

$$\{(x_1, u_1), \ldots\} \sim \mathrm{PP}(\rho_{\sigma_U,\tau_U} \times \mathrm{Leb}_{[0,s]}) \quad \theta_{ik} \overset{iid}{\sim} \mathrm{Gamma}(a, b)$$

- Sample item features:

$$\{(y_1, t_1), \ldots\} \sim \mathrm{PP}(\rho_{\sigma_I,\tau_I} \times \mathrm{Leb}_{[0,\alpha]}) \quad \beta_{jk} \overset{iid}{\sim} \mathrm{Gamma}(c, d)$$

- Given user features and item features:

$$e_{ij}^k | x_i, \{\theta_{ik}\}, y_j, \{\beta_{jk}\} \overset{ind}{\sim} \mathrm{Poi}(x_i y_j \theta_{ik} \beta_{jk})$$

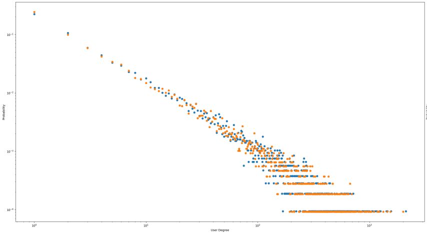
Probability of edge between user u_i and item t_j is given as $\Pr\{\sum_k e_{ij}^k > 0\}$

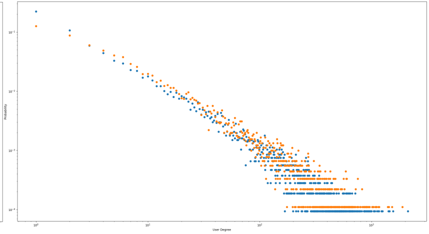
Variational Inference

- Model is defined in a way, using gamma-poisson conjugacy, complete conditionals for parameters lie in exponential family
- Use mean field approximation for the posterior distribution over parameters
- Variational distributions for all parameters are in the same exponential family as their complete conditionals
- We follow a coordinate ascent inference scheme described in [GHB14] for scaling inference
- (Forthcoming Paper for more details)

Posterior simulation

- We simulated a sparse graph of large size.
- Based on observation of a subgraph, we computed the posterior predictive distribution of the degree distribution of items for
  1. a sparse GGP prior (left figure), and
  2. a dense GGP prior (right figure).
- In both case the blue curve corresponds to the true degree distribution and the orange curve corresponds to the posterior predictive average.





Summary

- Described Sparse Exchangeable models for Machine Learning
- Sampling Insights:
  - Test for sparsity
  - New train-test split for relational data
- Extended Example: Sparse Non-Negative Matrix Factorization

Summary

- Described Sparse Exchangeable models for Machine Learning
- Sampling Insights:
  - Test for sparsity
  - New train-test split for relational data
- Extended Example: Sparse Non-Negative Matrix Factorization

That's all Folks!