Case Study 4: Collaborative Filtering

Review: Probabilistic Matrix Factorization

Probabilistic Matrix Factorization (PMF)

- A generative process:
  - Pick user factors $L_u \sim N(0, \sigma_u^2)$
  - Pick movie factors $R_v \sim N(0, \sigma_v^2)$
  - For each (user,movie) pair observed:
    - Pick rating as $L_u^T R_v + \text{noise}$

- Joint probability:
  $$P(L,R,X) = P(L) P(R) P(X|L,R)$$
PMF Graphical Model

\[ P(L, R \mid X) \propto P(L)P(R)P(X \mid L, R) \]

- Graphically:
  - posterior \( \propto \) joint prob.
  - ind. factors apriori
  - inf. these
  - ob. these
  - coupled in observation

MAP versus Regularized Least-Squares for Matrix Completion

- MAP under Gaussian Model:
  \[
  \max_{L,R} \log P(L, R \mid X) = \\
  - \frac{1}{2\sigma_u^2} \sum_u L_u^2 - \frac{1}{2\sigma_v^2} \sum_v R_v^2 - \frac{1}{2\sigma_r^2} \sum_{r_{uv}} (L_u \cdot R_v - r_{uv})^2 + \text{const}
  \]

- Least-squares matrix completion with L_2 regularization:
  \[
  \min_{L,R} \frac{1}{2} \sum_{u,v} (L_u \cdot R_v - r_{uv})^2 + \frac{\lambda_u}{2} ||L||_F^2 + \frac{\lambda_v}{2} ||R||_F^2
  \]

- Understanding as a probabilistic model is very useful! E.g.,
  - Change priors
  - Incorporate other sources of information or dependencies

©Emily Fox 2014
Posterior Computations

- MAP estimation focuses on point estimation:
  \[ \hat{\theta}_{MAP} = \arg \max_{\theta} p(\theta \mid x) \]

- What if we want a full characterization of the posterior?
  - Maintain a measure of uncertainty
  - Estimators other than posterior mode (different loss functions)
  - Predictive distributions for future observations

- Often no closed-form characterization (e.g., mixture models, PMF, etc.)

Bayesian PMF Example

- Latent user and movie factors:
  \[ L_u \sim N(m_u, \Sigma_u) \quad u = 1, \ldots, n \]
  \[ R_v \sim N(m_v, \Sigma_v) \quad v = 1, \ldots, m \]

- Observations

- Hyperparameters:

- Want to predict new movie rating:
  \[ p(r_{uv} \mid x, \phi) = \int p(r_{uv} \mid L_u, R_v) p(L_u, R_v \mid \phi) \, dL_u \, dR_v \]
Bayesian PMF Example

\[ p(r_{uv}^* \mid X, \phi) = \int p(r_{uv}^* \mid L_u, R_v)p(L, R \mid X, \phi)dLdR \]

- Monte Carlo methods:
  \[ p(r_{uv}^* \mid X, \phi) \approx \frac{1}{N} \sum_{k=1}^{N} p(r_{uv}^* \mid L_u^{(k)}, R_v^{(k)}) \]
  \[ \text{samples from posterior} \]
  \[ \text{Approx. as} \]

- Ideally:
  \[ p(L, R \mid X, \phi) \]
  \[ = \int p(L \mid X, R, \phi)p(R) \, dR \]
  \[ = \int \frac{p(X \mid L, R)p(L \mid \phi)}{p(X \mid \phi)} \, dR \]
  \[ \text{Again, intractable} \]

Bayesian PMF Gibbs Sampler

- Outline of Bayesian PMF sampler
  1. Init \( L^{(0)}, R^{(0)} \)
  2. For \( k=1, \ldots, \text{Niter} \)
     (i) Sample hyperparams \( \phi^{(k)}, \beta_h^{(k)}, \beta_r^{(k)}, \beta_{uv}^{(k)} \)
     (ii) For each user \( u=1, \ldots, n \) sample in parallel
         \[ L_u^{(k+1)} \sim p(L_u \mid X, R^{(k)}, \phi^{(k)}) \]
     (iii) For each movie \( v=1, \ldots, m \) sample in parallel
         \[ R_v^{(k+1)} \sim p(R_v \mid X, L^{(k+1)}, \phi^{(k)}) \]

Very similar to ideas of ALS (systematically)
Bayesian PMF Results

- Netflix data with:
  - Training set = 100,480,507 ratings from 480,189 users on 17,770 movie titles
  - Validation set = 1,408,395 ratings.
  - Test set = 2,817,131 user/movie pairs with the ratings withheld.

![Graph showing performance comparison between different models](image)

Figure 2. Left panel: Performance of SVD, PMF, logistic PMF, and Bayesian PMF using 30D feature vectors, on the Netflix validation data. The y-axis displays RMSE (root mean squared error), and the x-axis shows the number of epochs, or passes, through the entire training set. Right panel: RMSE for the Bayesian PMF models on the validation set as a function of the number of samples generated. The two curves are for the models with 30D and 60D feature vectors.

Bayesian PMF Results

- Bayesian model better controls for overfitting by averaging over possible parameters (instead of committing to one)

<table>
<thead>
<tr>
<th>D</th>
<th>Valid. RMSE PMF</th>
<th>BPMF Inc.</th>
<th>Test RMSE % PMF</th>
<th>BPMF Inc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>0.9154</td>
<td>0.8994</td>
<td>1.73</td>
<td>0.9188</td>
</tr>
<tr>
<td>40</td>
<td>0.9135</td>
<td>0.8968</td>
<td>1.83</td>
<td>0.9170</td>
</tr>
<tr>
<td>60</td>
<td>0.9150</td>
<td>0.8954</td>
<td>2.14</td>
<td>0.9185</td>
</tr>
<tr>
<td>150</td>
<td>0.9178</td>
<td>0.8931</td>
<td>2.69</td>
<td>0.9211</td>
</tr>
<tr>
<td>300</td>
<td>0.9231</td>
<td>0.8920</td>
<td>3.37</td>
<td>0.9265</td>
</tr>
</tbody>
</table>

Table 1. Performance of Bayesian PMF (BPMF) and linear PMF on Netflix validation and test sets.

Note: Each sampling stage of BPMF requires an $O(D^3)$ operation, so not for free.
What you need to know…

- Idea of full posterior inference vs. MAP estimation
- Gibbs sampling as an MCMC approach
- Example of inference in Bayesian probabilistic matrix factorization model

Case Study 4: Collaborative Filtering

Matrix Factorization and Probabilistic LFM for Network Modeling

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Emily Fox
February 20th, 2014
Network Data

- Structure of network data

Similarities to Netflix data:
- Matrix-valued data (adj. matrix)
- High-dimensional
- Sparse
- Few links between nodes (e.g., ppl in social networks)

Differences
- Square
- Binary

Properties of Data Source
Matrix Factorization for Network Data

- Vanilla matrix factorization approach:
  In undirected case, just introduce node (eg user) factors $L_u$ and $L_v$.
  $r_{uv} \sim L_u \cdot L_v \quad \text{edge bt users } u + v$

  In directed (or asymmetric) case, introduce sender factors $L_u$ and receiver factors $L_v$ (every node/user has both $L_u$ and $L_v$).
  $r_{uv} \sim L_u \cdot L_v \quad \text{edge from user } u \text{ to user } v$

- What to return for link prediction?
  Is $r_{uv}$ binary? $L_u \in \mathbb{R}^k \rightarrow \text{no.}$
  Many options, but can return top $K$
  $r_{uv}, \ldots, r_{uvK}$ (just use threshold rule)

- Slightly fancier:
  More appropriate to have $r_{uv} \in [0, 2]$.
  Use $r_{uv} = \sigma(L_u \cdot L_v)$, $\sigma = \text{logistic function}$

Probabilistic Latent Space Models

- Assume features (covariates) of the user $x_u$ or relationship $x_{uv}$.
- Each user has a “position” in a $k$-dimensional latent space $L_u \in \mathbb{R}^k$.

- Probability of link:
  \[
  \log \text{odds of } r_{uv} = 1 \mid L_u, L_v, x_{uv}, \beta \quad = \log \frac{P(r_{uv} = 1 \mid L_u, L_v, x_{uv}, \beta)}{P(r_{uv} = 0 \mid L_u, L_v, x_{uv}, \beta)}
  \]
  \[
  = \beta_0 + \beta_x^T x_{uv} - L_u \cdot L_v
  \]
  \[
  = \beta_0 + \beta_{x_{uv}} \cdot 0 - L_u \cdot L_v
  \]
**Probabilistic Latent Space Models**

- Probability of link:
  \[
  \log \text{odds } p(r_{uv} = 1 \mid L_u, L_v, x_{uv}, \beta) = \beta_0 + \beta^T x_{uv} - |L_u - L_v|
  \]

- Bayesian approach:
  - Place prior on user factors and regression coefficients
  - Place hyperprior on user factor hyperparameters
  - Many other options and extensions (e.g., can use GMM for clustering of users in the latent space)

\[
\log \text{odds } p(r_{uv} = 1 \mid L_u, L_v, x_{uv}, \beta) = \beta_0 + \beta^T x_{uv} + |L_u^T L_v|
\]

**What you need to know…**

- Representation of network data as a matrix
  - Adjacency matrix

- Similarities and differences between adjacency matrices and general matrix-valued data

- Matrix factorization approaches for network data
  - Just use standard MF and threshold output
  - Introduce link functions to constrain predicted values

- Probabilistic latent space models
  - Model link probabilities using distance between latent factors