Case Study 4: Collaborative Filtering

Probabilistic Matrix Factorization

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Sham Kakade
May 26, 2016

Matrix Completion and Local Minima

(J. Lee, K. Ge, T. M., yesterday) + another related paper on matrix sensing

with slightly stronger standard inc. assumption

no local min for non-convex form.
Cold-Start vs. Overfitting vs. Consistency?

- Suppose a user $u$ has $k$ ratings and there is noise (i.e. you observe the true matrix + noise).
  - How do we estimate $L_u$?
  - Suppose the number of people goes to infinity, can we estimate the $R_v$'s consistently?
    - Using ALS?
    - Instead:
      \[
      X = LR, \quad \frac{1}{m} X^T X = R M R^T
      \]

Probabilistic Matrix Factorization (PMF)

- A generative process:
  - Pick user $u$ factors
    \[
    L_u \sim N(0, \sigma_u^2)
    \]
  - Pick movie $v$ factors
    \[
    R_v \sim N(0, \sigma_v^2)
    \]
  - For each (user,movie) pair observed:
    - Pick rating as $L_u \cdot R_v +$ noise
      \[
      \sqrt{\nu} \sim N(L_u \cdot R_v, \sigma^2)
      \]
  - Joint probability:
    \[
    P(L, R, X) = P(L)P(R)P(X)LR
    \]
PMF Graphical Model

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

- Graphically:

Maximum A Posteriori for Matrix Completion

\[
P(L, R | X) \propto p(L)p(R)p(X | L, R)
\]
\[
\propto e^{-\frac{1}{2\sigma_u^2} \sum_{u=1}^{n} L_u^2} e^{-\frac{1}{2\sigma_v^2} \sum_{v=1}^{k} R_v^2} e^{-\frac{1}{2\sigma_f^2} \sum_{uv} (L_u \cdot R_v - r_{uv})^2}
\]

\[
\begin{align*}
\max_{L, R} & \quad P(L, R | X) \\
\text{subject to} & \quad L_u \geq 0, R_v \geq 0, u \in [n], v \in [k],
\end{align*}
\]
\[
\begin{align*}
\lambda_u & = \frac{\sigma_u^2}{\sigma_u^2} \quad \lambda_v = \frac{\sigma_v^2}{\sigma_v^2}
\end{align*}
\]

\[
\min_{L, R} \quad \frac{1}{2} ||X||_F^2 + \frac{\lambda_u}{2} ||L||_F^2 + \frac{\lambda_v}{2} ||R||_F^2 + \frac{1}{2} \sum_{u,v} (L_u \cdot R_v - r_{uv})^2
\]

\[
\begin{align*}
\text{if} & \quad \sigma_u \text{ same for } u, \\
& \quad \text{same } \lambda_u, \sigma_v \text{ same for } v.
\end{align*}
\]
MAP versus Regularized Least-Squares for Matrix Completion

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

- Least-squares matrix completion with L2 regularization:
  \[
  \min_{L,R} \frac{1}{2} \sum_{ruv} (L_u \cdot R_v - r_{uv})^2 + \frac{\lambda_u}{2} \|L\|_F^2 + \frac{\lambda_v}{2} \|R\|_F^2
  \]

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

What you need to know...

- Probabilistic model for collaborative filtering
  - Models, choice of priors
  - MAP equivalent to optimization for matrix completion

©Sham Kakade 2016
Case Study 4: Collaborative Filtering

Gibbs Sampling for Bayesian Inference

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:
  \[ \mathbf{L}_u \sim N(\mu_u, \Sigma_u) \quad \mathbf{R}_v \sim N(\mu_v, \Sigma_v) \quad \mathbf{t}_{uv} \sim N(\mathbf{L}_u \cdot \mathbf{R}_v, \sigma^2) \]

- Observations
- Hyperparameters:

- Want to predict new movie rating:
  \[ P(r_{uv} \mid X_{obs}, \phi) = \int P(r_{uv} \mid L_u, R_v) P(L_u, R_v \mid X_{obs}, \phi) dL_u dR_v \]

Bayesian PMF vs. MAP PMF

\[ p(r^*_{uv} \mid X, \phi) = \int p(r^*_{uv} \mid L_u, R_v) p(L, R \mid X, \phi) dL dR \]

- Relationship to MAP plug-in estimator:

MAP appr. uses:
\[ P(r^*_{uv} \mid X_{obs}, \phi) \approx \frac{1}{M} \sum_{m=1}^{M} p(r^*_{uv} \mid L_u, R_v) \]

MAP is approximating integral.
Bayesian PMF Example

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

- Monte Carlo methods:

  \[ p(L, R | X, \phi) \]

- Ideally:

  \[ p(L, R | X) = \frac{p(X | L, R) p(L) p(R)}{\int p(X | L, R) p(L) p(R) dL dR} \]

Bayesian PMF Example

- Want posterior samples

  \( (L^{(k)}, R^{(k)}) \sim p(L, R | X, \phi) \)

- What can we sample from?

  \[ p(L | X, R) \]

  \[ = \frac{\prod_{u \in V_{obs}} p(L_u | L, R, \phi) \prod_{v \in V_{obs}} p(r_{uv} | L_u, R_v)}{z} \]

  \[ = \frac{\prod_{u \in V_{obs}} [p(L_u) \prod_{v \in V_{obs}} p(r_{uv} | L_u, R_v)]}{z} \]
Bayesian PMF Example

- For user u:
  
  \[ p(L_u \mid X, R, \phi_u) \propto p(L_u \mid \phi_u) \prod_{v \in V_u} p(r_{uv} \mid L_u, R_v, \phi_r) \]

  \[ = N(\mu_{uv}, \Sigma_{uv}) \prod_{v \in V_u} N(r_{uv} \mid \mu_{uv}, \Sigma_{uv}) \]

  \[ \mu_{uv} = \sum_{v \in V_u} \Sigma_{uv}^{-1} \Sigma_{uv}^{-1} \mu_{uv} \]

  \[ \Sigma_{uv}^{-1} = \Sigma_{uv}^{-1} + \sigma_u^{-2} \sum_{v \in V_u} R_v R_v^T \]

- Symmetrically for \( R_v \) conditioned on \( L \) (breaks down over movies)
- Luckily, we can use this to get our desired posterior samples

Gibb Sampling

- Want draws:

  \[ (\Theta_1, \ldots, \Theta_n) = \Theta \]

  \[ (\Theta_1, \ldots, \Theta_n) \sim \Pi(\Theta) \leftarrow \text{joint} \]

  \[ (\Theta_1, \ldots, \Theta_n) \sim \prod_i \Pi(\Theta_i) \]

  \[ \Theta_1 \sim P_r(L_1 \mid X) \]

  \[ \ldots \]

  \[ \Theta_n \sim P_r(L_n \mid X) \]

  \[ \text{for } i = 1 \ldots n \]

- Construct Markov chain whose steady state distribution is
- Then, asymptotically correct

- Simplest case:

  \[ \Theta_w \sim P_r(\Theta_w \mid \Theta_1, \ldots, \Theta_n) \]

  \[ \text{for } w = 1 \ldots n \]

  \[ \Theta \text{ converges to } \Pi. \]
Bayesian PMF Gibbs Sampler

- Outline of Bayesian PMF sampler

\[
\begin{align*}
&1) \text{ int. } C \sim \eta_{i\ell} \\
&2) \text{ for each user } i = 1, \ldots, n \text{ sample } \eta_{i\ell} \sim p(\eta_{i\ell}|C(\eta_{i\ell})) \\
&\text{ for each movie } v = 1, \ldots, m \text{ sample } \mu_{v\ell} \sim p(\mu_{v\ell}|C(\mu_{v\ell})) \\
&\text{ (similarity by ALS)}
\end{align*}
\]

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.

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

From Salakhutdinov and Mnih, ICML 2008

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</th>
<th>% Inc.</th>
<th>Test RMSE</th>
<th>% Inc.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PMF</td>
<td>BPMF</td>
<td>PMF</td>
<td>BPMF</td>
</tr>
<tr>
<td>30</td>
<td>0.9154</td>
<td>0.8994</td>
<td>1.74</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.

Bayesian PMF Gibbs Sampler

Outline of Bayesian PMF sampler
1. Initialize $L^{(1)}$, $R^{(1)}$
2. For $k = 1, \ldots, N_{iter}$
   (i) Sample hyperparams $\phi^{(k)}$
   (ii) For each user $u = 1, \ldots, n$ sample (in parallel)
        $L_u^{(k+1)} \sim N(\hat{\mu}_u, \hat{\Sigma}_u)$
   (iii) For each move $v = 1, \ldots, m$ sample (in parallel)
         $R_v^{(k+1)} \sim N(\hat{\mu}_v, \hat{\Sigma}_v)$

where $\hat{\Sigma}_u^{-1} = \Sigma_u^{-1} + \sigma^-2 \sum_{v \in V_u} R_v R_v^T$

\[
\hat{\mu}_u = \hat{\Sigma}_u \left( \sigma^{-2} \sum_{v \in V_u} r_{uv} R_v + \Sigma_u^{-1} \mu_u \right)
\]
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
• Implementation of vanilla sampler in GraphLab

Case Study 4: Collaborative Filtering

Matrix Factorization and Probabilistic LFM's for Network Modeling

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Sham Kakade
May 26, 2016
Network Data

- Structure of network data

Properties of Data Source

- Similarities to Netflix data:
  - Matrix
  - High-dimensional
  - Sparse

- Differences
  - Square
  - Binary
Matrix Factorization for Network Data

- Vanilla matrix factorization approach:

  - What to return for link prediction?

- Slightly fancier:

Algorithms
Algorithms

Probabilistic Latent Space Models

- Assume features (covariates) of the user or relationship
- Each user has a “position” in a $k$-dimensional latent space

- Probability of link:
Probabilistic Latent Space Models

- Probability of link:

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

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

- 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 \( L_u \) → clustering of users in the latent space)

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