Quick Recap
Scaling Neighbourhood Methods
Collaborative Filtering

- \( m = \#\text{items} \)
- \( n = \#\text{users} \)
- Complexity : \( m \times m \times n \)
Comparative Scale of Signals

- ~50 M users
- ~25 M items
- Explicit Ratings ~ $O(1M)$ (1 per billion)
- Purchase ~ $O(100M)$ (100 per billion)
- Browse ~ $O(10B)$ (10000 per billion)
Implicit Signals Used

- Bought History
- Browse History
- Compare History
Category-partitioned v/s Category independent
KEEP CALM AND CARRY ON
KEEP CALM AND CARRY ON
Similarity Metric for boolean matrix

• Cosine Similarity

  A. Pair Count (p) - P1 and P2

  B. Individual Count (n_i) - P1, P2 individually
Empowering Map Reduce

- Calculate 'p' by forming pairs and counting
- Calculate 'n1' by making P1 as the key
- Calculate 'n2' by making P2 as the key
- Took 2 hours on 5 years of data
- Scales Horizontally
Map Reduce 1

B1 -> P1, P2, P3, P4,
B2 -> P2, P3, P4, P5,
Generating pairs:

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1 P2</td>
<td>1</td>
</tr>
<tr>
<td>P1 P3</td>
<td>1</td>
</tr>
<tr>
<td>P1 P4</td>
<td>1</td>
</tr>
<tr>
<td>P2 P3</td>
<td>1</td>
</tr>
<tr>
<td>P2 P4</td>
<td>1</td>
</tr>
</tbody>
</table>

Mapper: Key(Pair of items) => Value(weight)
Reducer: Accumulates the weights for each Pair.

Reducer O/P

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1 P2</td>
<td>1</td>
</tr>
<tr>
<td>P1 P3</td>
<td>1</td>
</tr>
<tr>
<td>P1 P4</td>
<td>1</td>
</tr>
<tr>
<td>P2 P3</td>
<td>2</td>
</tr>
<tr>
<td>P2 P4</td>
<td>2</td>
</tr>
<tr>
<td>P2 P5</td>
<td>1</td>
</tr>
<tr>
<td>P3 P4</td>
<td>1</td>
</tr>
<tr>
<td>P3 P5</td>
<td>1</td>
</tr>
<tr>
<td>P4 P5</td>
<td>1</td>
</tr>
</tbody>
</table>
Calculating the value 'n1':

Input:
- P1 P4 1
- P2 P3 1
- P4 P5 1
- P2 P4 1

Mapper: Key (i1) => Value (Pairs with weights)
Reducer: Accumulates the i1's to form the Pairs with weights, n1

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>P1 P4 1 1</td>
</tr>
<tr>
<td>P2</td>
<td>P2 P3 1 1</td>
</tr>
<tr>
<td>P4</td>
<td>P4 P5 1 1</td>
</tr>
<tr>
<td>P2</td>
<td>P2 P4 1 1</td>
</tr>
</tbody>
</table>

Reducer Output

| P1 P4 1 1 |
| P2 P3 1 2 |
| P4 P5 1 1 |
| P2 P4 1 2 |
Calculating the value 'n2':

Input:

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>P4 1 1</td>
</tr>
<tr>
<td>P2</td>
<td>P3 1 1</td>
</tr>
<tr>
<td>P4</td>
<td>P5 1 1</td>
</tr>
<tr>
<td>P2</td>
<td>P4 1</td>
</tr>
</tbody>
</table>

Mapper: Key (i2) => Value(Pairs with weights)

Reducer: Accumulates the i2's to form the Pairs with weights, n2

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Reducer Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>P4</td>
<td>P1 P4 1 1 1</td>
<td>P1 P4 1 1 2</td>
</tr>
<tr>
<td>P3</td>
<td>P2 P3 1 1 1</td>
<td>P2 P3 1 2 1</td>
</tr>
<tr>
<td>P5</td>
<td>P4 P5 1 1 1</td>
<td>P4 P5 1 1 1</td>
</tr>
<tr>
<td>P4</td>
<td>P2 P4 1 1 1</td>
<td>P2 P4 1 2 2</td>
</tr>
</tbody>
</table>
Latent Variable Models and Factorization Models

Akash Khandelwal, Avijit Saha, Mohit Kumar, Vivek Mehta
Outline

- Introduction
  - Recommender Systems Recap
  - Latent Variable Models
  - Factorization Models
- Matrix Factorization
  - Singular Value Decomposition
  - BPMF
- Factorization Machine
- Conclusion
Recommender Systems (RSs)

- Collaborative Filtering (CF)
  - Neighborhood Based
    - KNN
  - Model Based
    - Cluster-based CF and Bayesian classifiers.
    - Latent variable models such as, LDA, pLSA, and matrix factorization (MF).
- Content Based
- Knowledge Based
- Hybrid
Latent Variable Models

- Supplementing a set of observed variables with additional latent, or hidden, variables.
- Latent variable models are widely used in several domains such as machine learning, statistics, data mining.
- Reveals hidden structure which explains the data.
- Latent variable models consider a joint distribution over the hidden and observed variables.
- Hidden structure is found by calculating the posterior.
- LDA is an example of latent variable models.
Factorization Models

- One of the widely used latent variable models in the RSs community.
- Preferences of a user are determined by a small number of unobserved latent factors.
  - Matrix Factorization: Each user and item are mapped to a latent factor vector:
    \[ u_i \in \mathbb{R}^K \]
    \[ v_j \in \mathbb{R}^K \]
  - Tensor Factorization: Mapping of each variable of each category type to a \( K \) dimensional latent factor vector.
- Many problem specific factorization models.
Focus

- Factorization Models
  - Matrix Factorization (MF)
  - Factorization Machine (FM)
Singular value decomposition (SVD) is a factorization of a matrix. Formally, the SVD of an $R \in \mathbb{R}^{I \times J}$ is:

$$ R = U\Sigma V^*, $$

where,

- $U = I \times I$ unitary matrix
- $\Sigma = I \times J$ rectangular diagonal matrix
- $V^* = J \times J$ unitary matrix

$\sigma_{i,i}$ of $\Sigma$ are known as the singular values of $R$

$I$ columns of $U$ and the $J$ columns of $V$ are called the left-singular vectors and right-singular vectors of $R$, respectively.
Let, $\mathbf{R} \in \mathbb{R}^{I \times J}$

Apply SVD: 

$$ R = U \Sigma V^* , $$(2)

Estimate: 

$$ \hat{R} = U \begin{pmatrix} \sigma_{1,1} & & \\ & \sigma_{2,2} & \\ & & \ddots \end{pmatrix} V^*. $$

$$ (3) $$
Example of SVD

(a)  
(b)  
(c)  
(d)
Figure 1: Matrix Factorization
Consider a user-movie matrix $R \in \mathbb{R}^{I \times J}$ where the $r_{ij}$ cell represents the rating provided to the $j^{th}$ movie by the $i^{th}$ user. MF decomposes the matrix $R$ into two low-rank matrices $U = [u_1, u_2, ..., u_I]^T \in \mathbb{R}^{I \times K}$ and $V = [v_1, v_2, ..., v_J]^T \in \mathbb{R}^{J \times K}$:

$$R \sim UV^T. \quad (4)$$

$$\sum_{(i,j) \in \Omega} \left( r_{ij} - u_i^T v_j \right)^2 \quad (5)$$
SVD with $K$ singular value would been the solution if $R$ is fully observed.

However, $R$ is partially observed.

Solution: Stochastic gradient descent to rank-1 update:

\[
e_{ij} = r_{ij} - u_i^T v_j
\]

\[
u_{ik} = u_{ik} + \nu (e_{ij} v_{jk} - \lambda u_{ik}) \tag{6}
\]

\[
v_{jk} = v_{jk} + \nu (e_{ij} u_{ik} - \lambda v_{jk}) \tag{7}
\]

Then iterate this for each rank $K$. 

Problem with SVD approach

- Learning rate and regularization parameters needs to be tuned manually.
- Overfitting.
- Solution: Bayesian Probabilistic Matrix Factorization (BPMF) [1].
The likelihood term of BPMF is as follows:

$$p(R | \Theta) = \prod_{(i,j) \in \Omega} \mathcal{N}(r_{ij} | u_i^T v_j, \tau^{-1}),$$

where $u_i$ is the latent factor vector for the $i^{\text{th}}$ user, $v_j$ is the latent factor vector for the $j^{\text{th}}$ item, $\tau$ is the model precision, $\Omega$ is the set of all observations, and $\Theta$ is the set of all the model parameters.
Independent priors are placed on all the model parameters in $\Theta$ as:

$$p(\mathbf{U}) = \prod_{i=1}^{l} \mathcal{N}(u_i | \mu_u, \Lambda_u^{-1}),$$  \hspace{1cm} (9)$$

$$p(\mathbf{V}) = \prod_{j=1}^{J} \mathcal{N}(v_j | \mu_v, \Lambda_v^{-1}).$$  \hspace{1cm} (10)$$
Further place Normal-Wishart priors are placed on all the hyperparameters $\Theta_H = \{\{\mu_u, \Lambda_u\}, \{\mu_v, \Lambda_v\}\}$ as:

\[
p(\mu_u, \Lambda_u) = \mathcal{NW}(\mu_u, \Lambda_u | \mu_0, \beta_0, W_0, \nu_0), \quad (11)
\]

\[
= \mathcal{N}(\mu_u | \mu_0, (\beta_0 \Lambda_u)^{-1}) \mathcal{W}(\Lambda_u | W_0, \nu_0)
\]

\[
p(\mu_v, \Lambda_v) = \mathcal{NW}(\mu_v, \Lambda_v | \mu_0, \beta_0, W_0, \nu_0). \quad (12)
\]

where

\[
\mathcal{W}(\Lambda | W_0, \nu_0) = \frac{1}{C} |\Lambda|^{\nu_0-D-1} \exp(-\frac{1}{2} Tr(W_0^{-1} \Lambda))
\]
Joint Distributions

The joint distribution of the observations and the hidden variables can be written as:

\[ p(R, \Theta, \Theta_H | \Theta_0) = p(R | \Theta)p(U)p(V)p(\mu_u, \Lambda_u | \Theta_0)p(\mu_v, \Lambda_v | \Theta_0), \]

(13)

where \( \Theta_0 = \{ \mu_0, \beta_0, W_0, \nu_0 \} \)
Inference

- Evaluation of the joint distribution in Eq. (13) is intractable.
- However, all the model parameters are conditionally conjugate.
- So we Gibbs sampler has closed form updates.
- Replacing Eq. (8)-(12) in Eq. (13), the sampling distribution of $u_i$ can be written as follows:

$$p(u_i|\cdot) \sim \mathcal{N}(u_i|\mu^*,(\Lambda^*)^{-1}), \quad (14)$$

$$\Lambda^* = \left(\begin{array}{c} \Lambda_u + \tau \sum_{j \in \Omega_i} v_j v_j^T \end{array}\right) \quad (15)$$

$$\mu^* = (\Lambda^*)^{-1} \left(\begin{array}{c} \Lambda_u \mu_u + \tau \sum_{j \in \Omega_i} v_j r_{ij} \end{array}\right) \quad (16)$$
Figure 2: RMSE vs Iterations
Factorization Models

- Matrix factorization [2].
- Tensor factorization [3].
- Specific models like SVD++ [4], TimeSVD++ [5], FPMC [6], and BPTF [3], etc. have been developed.
- Several Learning technique like SGD, ALS, variational Bayes, MCMC Gibbs sampling have been developed.
Advantages of Factorization Model:

- Scalability and Performance.

Problem:

- deriving inference techniques for each individual model is a time consuming task and requires considerable expertise.

Advantages of Feature Based Techniques:

- Generic approach.
- Can be solved using standard tools like LIBSVM or SVMLight.

Problem:

- Can not handle very sparse data.
Combines advantages of both factorization models and feature based model.
Can subsume many state of the art factorization model like SVD++, TimeSVD++, FPMC, PITF, etc.
Performs well for sparse data where SVMs fails.
Example: (U1, M1, G1, 2), (U1, M3, G2, 5), (U2, M2, G3, 4) and (U3, M1, G1, 5)

<table>
<thead>
<tr>
<th>User</th>
<th>Feature x</th>
<th>Target y</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>1 0 0</td>
<td>2 y1</td>
</tr>
<tr>
<td>x2</td>
<td>1 0 0</td>
<td>5 y2</td>
</tr>
<tr>
<td>x3</td>
<td>0 1 0</td>
<td>4 y3</td>
</tr>
<tr>
<td>x4</td>
<td>0 0 1</td>
<td>5 y4</td>
</tr>
<tr>
<td>U1</td>
<td>M1</td>
<td>G1</td>
</tr>
<tr>
<td>U2</td>
<td>M2</td>
<td>G2</td>
</tr>
<tr>
<td>U3</td>
<td>M3</td>
<td>G3</td>
</tr>
</tbody>
</table>

Figure 3: Factorization Machine
Following is the equation for FM.

\[ y_n = w_0 + \sum_{i=1}^{D} w_i x_{ni} + \sum_{i=1}^{D} \sum_{j=i+1}^{D} x_{ni} x_{nj} \sum_{k=1}^{K} v_{ik} v_{jk} \]  

(17)

Assumptions of FM are following:

\[ y_n | x_n, \theta \sim \mathcal{N}(\hat{y}(x_n, \theta), \alpha^{-1}) \]

\[ y_n | x_n, \theta \sim \text{Bernoulli}(b(\hat{y}(x_n, \theta))) \]

And L2 regularization on \( \theta \)
\[ \hat{y} = w_0 + w_u + w_i + \mathbf{v}_u^T \mathbf{v}_i \]  

(18)

Feature representation of FM: 
\[ D = |U \cup I| \]

\[ x_j = \delta(j = u \lor j = i) \]
Mimic SVD++

\[
\hat{y} = w_0 + w_u + w_i + v_u^T v_i + \frac{1}{\sqrt{N}} \sum_{l \in N_u} v_i^T v_l
\]  

(19)

Feature representation of FM: \( D = |U \cup I \cup L| \)

\( x_j = 1 \) if \( j = u \lor j = i \)

\[ = \frac{1}{\sqrt{N}} \] if \( j \in N_u \)

\[ = 0 \] else
- Stochastic gradient descent is the simplest algorithm to solve FM (SGD-FM).
- MCMC based Bayesian Factorization Machine gives state-of-the-art performance (MCMC-FM).
- Alternating least square and adaptive stochastic gradient descent.
Comparison of each Leaning Techniques

- SGD-FM
  - Pros: SGD online algorithm and more scalable.
  - Cons: Costly cross validation of parameters.

- MCMC-FM
  - Pros: Performance.
  - No cross validation required.
Following is the equation for FM.

\[
\hat{y}_n = w_0 + \sum_{i=1}^{D} w_i x_{ni} + \sum_{i=1}^{D} \sum_{j=i+1}^{D} x_{ni} x_{nj} \sum_{k=1}^{K} v_{ik} v_{jk}
\]  

(20)

Cost function is:

\[
\sum_{n=1}^{N} (y_n - \hat{y}_n)^2 + L2 \text{ regularization}
\]

(21)

SGD update equations:

\[
v_{ik} = v_{ik} + \nu \left( 2 \cdot (y_n - \hat{y}_n) x_{ni} \sum_{l=1 \& l \neq i}^{N} x_{nl} v_{nl} - 2\lambda v_{ik} \right)
\]

(22)
Likelihood:

\[ y_n \sim \mathcal{N}(y_n | \hat{y}_n, \tau) \]

Prior:

\[ p(w_0) \sim \mathcal{N}(w_0 | \mu_0, \sigma_0) \]
\[ p(w_i) \sim \mathcal{N}(w_i | \mu_w, \sigma_w) \]
\[ p(v_{ik}) \sim \mathcal{N}(v_{ik} | \mu_k, \sigma_k) \]

Hyperprior:

\[ p(\mu_w) \sim \mathcal{N}(\mu_w | \mu, \sigma_w \nu_0) \quad p(\sigma_w) \sim G(\sigma_w | \alpha_0, \beta_0) \]
\[ p(\mu_k) \sim \mathcal{N}(\mu_k | \mu, \sigma_k \nu_0) \quad p(\sigma_k) \sim G(\sigma_k | \alpha_0, \beta_0) \]
Results

Figure 4: RMSE vs Iterations
Datasets

- Yelp challenge
- Yelp datasets
- Users information
- Social information
- Location
- Time
- Ratings
- Reviews
References


THANK YOU