Non-negative matrix factorization

Mathieu Blondel
NTT Communication Science Laboratories
2014/10/28
Outline

- Non-negative matrix factorization (NMF)
- Optimization algorithms
- Passive-aggressive algorithms for NMF
Non-negative matrix factorization
Non-negative matrix factorization (NMF)

Given observed matrix $R \in \mathbb{R}^{n \times d}$, find matrices $P \in \mathbb{R}_{+}^{n \times m}$ and $Q \in \mathbb{R}_{+}^{m \times d}$ such that

$$R \approx PQ$$

$$\begin{bmatrix} r_{1,1} & \cdots & r_{1,d} \\ \vdots & \ddots & \vdots \\ r_{n,1} & \cdots & r_{n,d} \end{bmatrix} \approx \begin{bmatrix} p_{1,1} & \cdots & p_{1,m} \\ \vdots & \ddots & \vdots \\ p_{n,1} & \cdots & p_{n,m} \end{bmatrix} \times \begin{bmatrix} q_{1,1} & \cdots & q_{1,d} \\ \vdots & \ddots & \vdots \\ q_{m,1} & \cdots & q_{m,d} \end{bmatrix}$$

$m$ is a user-given hyper-parameter

$PQ$ is called a **low-rank approximation** of $R$
Examples of non-negative data

The matrix $R$ could contain...

- Number of word occurrences in text documents
- Pixel intensities in images
- Ratings given by users to movies
- Magnitude spectrogram of an audio signal
- etc...
Why imposing non-negativity of $P$ and $Q$?

- Natural assumption if $R$ is non-negative

- Each row of $R$ is approximated by a **strictly additive** combination of factors / bases / atoms

$$[r_{u,1}, \cdots, r_{u,d}] \approx \sum_{k=1}^{m} p_{u,k} \times [q_{k,1}, \cdots, q_{k,d}]$$

- $P$ and $Q$ tend to be sparse (have many zeros) 
  $\Rightarrow$ easy-to-interpret, part-based solution
Application 1: document analysis

- $R$ is a collection of $n$ text documents
- Each row $[r_{u,1}, \ldots, r_{u,d}]$ of $R$ corresponds to a document represented as a bag of words
- $r_{u,i}$ is the number of occurrences of word $i$ in document $u$
- Factors $[q_{k,1}, \ldots, q_{k,d}]$ in $Q$ correspond to “topics”
- $p_{u,k}$ is the weight of topic $k$ in document $u$
Application 1: document analysis

Using \( n = 30,991 \) articles from Grolier encyclopedia, vocabulary size \( d = 15,276 \) and number of topics \( m = 200 \) [Lee & Seung, 99]
Application 2: image processing

- $R$ is a collection of $n$ images or image patches
- Each row $[r_{u,1}, \cdots, r_{u,d}]$ of $R$ corresponds to an image or image patch
- $r_{u,i}$ is the pixel intensity of pixel $i$ in image $u$
- Factors $[q_{k,1}, \cdots, q_{k,d}]$ in $Q$ correspond to image “parts”
- $p_{u,k}$ is the weight of part $k$ in image $u$
  ⇒ can be used as high-level feature descriptor
Application 2: image processing

Using $n = 2,429$ face images, $d = 19 \times 19$ pixels and $m = 49$ basis images [Lee & Seung, 99]
**Application 3: recommendation systems**

- $R$ is a **partially observed** rating matrix

![Rating Matrix](image)

In case of ratings, $r$ can be represented as a matrix $R$, as depicted in figure 2.1.

![Example of rating matrix](image)

In that case, the recommendation problem boils down to predict unknown values of $R$. The set of ratings given by some user $u$ will be represented by an incomplete array $R(u)$, while the rating of $u$ on some item $i$ will be denoted $R(u, i)$. Note that this value may be unknown. The subset of items $i \in I$ actually rated by $u$ is $I(u)$. The number of items in that set is $|I(u)|$. Similarly, the set of ratings given to some item $i$ will be represented by an incomplete array $R(i)$. The subset of users $u \in U$ which have actually rated $i$ is noted $U(i)$. The number of items in that set is $r_{u,i}$.
Application 3: recommendation systems

- Users and movies are projected in a common $m$-dimensional latent space [Louppe, 2010]
Application 3: recommendation systems

- Inner product in this space can be used to predict missing values

\[ r_{u,i} \approx \left[ p_{u,1}, \ldots, p_{u,m} \right] \begin{bmatrix} q_{1,i} \\ \vdots \\ q_{m,i} \end{bmatrix} \]
Optimization algorithms
Formulating an optimization problem

How do we find $P \in \mathbb{R}^{n \times m}$ and $Q \in \mathbb{R}^{m \times d}$ such that

$$R \approx PQ$$
Formulating an optimization problem

Using Euclidean distance:

\[
\begin{align*}
\text{minimize} \quad & F(P, Q) = \frac{1}{2} \| R - PQ \|^2 + \frac{\lambda}{2} \left( \| P \|^2 + \| Q \|^2 \right) \\
\text{subject to} \quad & P \geq 0, Q \geq 0
\end{align*}
\]

- **error term**
- **regularization term**

Non-convex in \( P \) and \( Q \) **jointly**

Convex in \( P \) or \( Q \) **separately**

\( \Rightarrow \) we can alternate between updating \( P \) and \( Q \)
Formulating an optimization problem

Using generalized KL divergence, a.k.a. I-divergence:

\[
\text{minimize } \begin{bmatrix} P \geq 0, Q \geq 0 \end{bmatrix} \quad F(P, Q) = D_I(R \| PQ) + \lambda \left( \| P \|^2 + \| Q \|^2 \right)
\]

where \( D_I(A \| B) = \sum_{u,i} A_{u,i} \log \left( \frac{A_{u,i}}{B_{u,i}} \right) - A_{u,i} + B_{u,i} \).

When \( \lambda = 0 \), equivalent to MLE solution assuming \( r_{u,i} \sim \text{Poisson}((PQ)_{u,i}) \) [Févotte, 2009]
Two kinds of sparsity

- Sparsity of non-zero entries
  \[ R = \begin{bmatrix} 1 & 0 & 3 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \]

- Sparsity of observed values
  \[ R = \begin{bmatrix} 1 & ? & 3 \\ ? & 2 & ? \\ ? & ? & 1 \end{bmatrix} \]

- These two settings require different algorithm design and implementation
Multiplicative method

- Euclidean distance, no regularization [Lee & Seung, 2001]

\[ P_{u,k} \leftarrow P_{u,k} \frac{(RQ^T)_{u,k}}{(PQQ^T)_{u,k}} \]

\[ Q_{k,i} \leftarrow Q_{k,i} \frac{(P^T R)_{k,i}}{(P^T P Q)_{k,i}} \]

- Similar updates for generalized KL divergence case

- Guarantees that the objective is non-increasing... [Lee & Seung, 2001]

- ...but not convergence [Lin, 2007]
Projected gradient method

- Gradient step followed by a truncation [Lin, 2007]

\[
P \leftarrow \max \left( P - \eta \nabla_P F(P, Q), 0 \right)
\]
\[
Q \leftarrow \max \left( Q - \eta \nabla_Q F(P, Q), 0 \right)
\]

- \( \eta \) can be fixed to a small constant or adjusted by line search

- Converges to a stationary point of \( F \)
Projected stochastic gradient method

- Objective with missing values:

\[
\minimize_{P \geq 0, Q \geq 0} F(P, Q) = \frac{1}{2|\Omega|} \sum_{(u, i) \in \Omega} (r_{u,i} - p_u \cdot q_i)^2 + \frac{\lambda}{2} \left( \|P\|^2 + \|Q\|^2 \right)
\]

where \( \Omega \) is the set of observed values
Projected stochastic gradient method

• Similar to projected gradient method but use a stochastic approximation of the gradient

\[ p_u \leftarrow \max \left( p_u - \eta \nabla_P^{(u,k)} F(P, Q), 0 \right) \]

\[ q_i \leftarrow \max \left( q_i - \eta \nabla_Q^{(k,i)} F(P, Q), 0 \right) \]

• Slow convergence in terms of number of iterations...

• ...but very low iteration cost
  \[ \Rightarrow \] very fast in practice 😊

• However, quite sensitive to the choice of \( \eta \) 😞
Coordinate descent

- Update a **single** variable at a time [Hsieh & Dhillon, 2011, Yu et al. 2012]

\[
P_{u,k} \leftarrow P_{u,k} + \arg\min_{\delta} F(P + E_{u,k}\delta, Q) \quad \text{or} \\
Q_{k,i} \leftarrow Q_{k,i} + \arg\min_{\delta} F(P, Q + E_{k,i}\delta)
\]

where \(E_{u,k}\) is a matrix with all elements zero except the \((u, k)\) element which equals one

- Closed-form update in the Euclidean distance case

- My personal favorite in the batch setting 😊

---

23 / 44
Passive-aggressive algorithms for NMF
Online algorithms

- In real-world applications, missing entries in $R$ may be observed in real time
  - A user gave a rating to a movie
  - A user clicked on a link
- Ideally, $P$ and $Q$ should be updated in real time to reflect the knowledge that we gained from the new entry
Online algorithms

1. Initialize $P$ and $Q$ randomly

\[
\begin{bmatrix}
q_{1,1} & q_{1,2} & q_{1,3} \\
q_{2,1} & q_{2,2} & q_{2,3} \\
q_{3,1} & q_{3,2} & q_{3,3}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot
\end{bmatrix}
\]
Online algorithms

2. An element of $R$ is revealed (e.g., a user rated a movie)

$$
\begin{bmatrix}
q_{1,1} & q_{1,2} & q_{1,3} \\
q_{2,1} & q_{2,2} & q_{2,3} \\
q_{3,1} & q_{3,2} & q_{3,3}
\end{bmatrix}
\begin{bmatrix}
\begin{bmatrix}
p_{1,1} & p_{1,2} & p_{1,3} \\
p_{2,1} & p_{2,2} & p_{2,3} \\
p_{3,1} & p_{3,2} & p_{3,3}
\end{bmatrix}
\end{bmatrix}
\begin{bmatrix}
? & ? & ? \\
? & ? & 3 \\
? & ? & ?
\end{bmatrix}
$$
3. Update corresponding row of $P$ and column of $Q$

$$
\begin{bmatrix}
p_{1,1} & p_{1,2} & p_{1,3} \\
p_{2,1} & p_{2,2} & p_{2,3} \\
p_{3,1} & p_{3,2} & p_{3,3}
\end{bmatrix}
\begin{bmatrix}
q_{1,1} & q_{1,2} & q_{1,3} \\
q_{2,1} & q_{2,2} & q_{2,3} \\
q_{3,1} & q_{3,2} & q_{3,3}
\end{bmatrix}
\begin{bmatrix}
? & ? & ? \\
? & ? & 3 \\
? & ? & ?
\end{bmatrix}
$$
Large-scale learning using online algorithms

- Online algorithms can also be used in a large-scale batch setting
- Online to batch conversion: make several passes over the dataset
- Advantages of online algorithms
  - Low iteration cost
  - Low memory footprint
  - Ease of implementation
Passive-aggressive algorithms for NMF

- Passive-aggressive [Crammer et al., 2006] are online algorithms for classification and regression
- Very popular in the Natural Language Processing (NLP) community
- We propose passive-aggressive algorithms for NMF
Passive-aggressive algorithms for NMF

- On iteration $t$, $r_{u_t,i_t}$ is revealed

- We propose to update $p_{u_t}$ and $q_{i_t}$ by

  $$p_{u_t}^{t+1} = \arg\min_{p \in \mathbb{R}_+^m} \frac{1}{2} \| p - p_{u_t}^t \|^2 \text{ s.t. } | p \cdot q_{i_t}^t - r_{u_t,i_t} | = 0$$

  $$q_{i_t}^{t+1} = \arg\min_{q \in \mathbb{R}_+^m} \frac{1}{2} \| q - q_{i_t}^t \|^2 \text{ s.t. } | p_{u_t}^t \cdot q - r_{u_t,i_t} | = 0$$

- Conservative (do not change model too much) and corrective (satisfy constraint) update
Passive-aggressive algorithms for NMF

Since the two problems are the same, we can simplify notation

\[ w = p \] or \[ w = q \] (variable)

\[ w_{t+1} = p_{u_t}^{t+1} \] or \[ w_{t+1} = q_{i_t}^{t+1} \] (solution)

\[ w_t = p_{u_t}^t \] or \[ w_t = q_{i_t}^t \] (current iterate)

\[ x_t = q_{i_t}^t \] or \[ x_t = p_{u_t}^t \] (input)

\[ y_t = r_{u_t,i_t} \] (target)
Passive-aggressive algorithms for NMF

• Allow to not perfectly fit the target

\[
\mathbf{w}_{t+1} = \arg\min_{\mathbf{w} \in \mathbb{R}^m_+} \frac{1}{2} \| \mathbf{w} - \mathbf{w}_t \|^2 \quad \text{s.t.} \quad |\mathbf{w} \cdot \mathbf{x}_t - y_t| \leq \epsilon
\]

• If \( |\mathbf{w}_t \cdot \mathbf{x}_t - y_t| \leq \epsilon \), the algorithm is “passive”, i.e., \( \mathbf{w}_{t+1} = \mathbf{w}_t \)

• Otherwise, it is “aggressive”: the model is updated
Passive-aggressive algorithms for NMF

- The previous update changes the model as much as needed to satisfy the constraint ⇒ potential overfitting

- Introduce a slack variable to allow some error

\[ w_{t+1}, \xi^* = \arg\min_{w \in \mathbb{R}^m_+, \xi \in \mathbb{R}_+} \frac{1}{2} \|w - w_t\|^2 + C\xi \]

s.t. \[ |w \cdot x_t - y_t| \leq \epsilon + \xi, \]

- \( C > 0 \) controls the trade-off between being conservative and corrective
Passive-aggressive algorithms for NMF

- The solution is of the form

\[ \mathbf{w}_{t+1} = \max \left( \mathbf{w}_t + (\kappa - \theta) \mathbf{x}_t, 0 \right) \]

where \( \kappa \) and \( \theta \) are non-negative scalars

- In our AISTATS paper, we present three \( O(m) \) methods for finding \( \kappa \) and \( \theta \) [Blondel et al., 2014]
  - An exact method
  - A bisection method
  - An approximate update method
Passive-aggressive algorithms for NMF

Difference between the effect of $\epsilon$ and $C$

- Increasing $\epsilon$ increases the number of passive updates
  $\Rightarrow$ trades some error for faster training

- Reducing $C$ reduces update aggressiveness, since $0 \leq \kappa \leq C$ and $0 \leq \theta \leq C$
  $\Rightarrow$ reduces overfitting
### NMF algorithm comparison

<table>
<thead>
<tr>
<th>Solver</th>
<th>Iteration cost</th>
<th>Online</th>
<th>Hyper-parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiplicative</td>
<td>😞</td>
<td>😞</td>
<td>☻</td>
</tr>
<tr>
<td>Projected grad.</td>
<td>😞</td>
<td>😞</td>
<td>☻</td>
</tr>
<tr>
<td>Projected stochastic grad.</td>
<td>☻</td>
<td>☻</td>
<td>😞</td>
</tr>
<tr>
<td>Coordinate descent</td>
<td>☻</td>
<td>😞</td>
<td>☻</td>
</tr>
<tr>
<td>Passive-Aggressive</td>
<td>☻</td>
<td>☻</td>
<td>☻</td>
</tr>
</tbody>
</table>

In the setting with missing values.
Experimental results

- Datasets used

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Users</th>
<th>Items</th>
<th>Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movielens10M</td>
<td>69,878</td>
<td>10,677</td>
<td>10,000,054</td>
</tr>
<tr>
<td>Netflix</td>
<td>480,189</td>
<td>17,770</td>
<td>100,480,507</td>
</tr>
<tr>
<td>Yahoo-Music</td>
<td>1,000,990</td>
<td>624,961</td>
<td>252,800,275</td>
</tr>
</tbody>
</table>

- We split ratings into 4/5 for training and 1/5 for testing

- The task is to predict ratings in the test set
Convergence results

Results w.r.t. test data on the MovieLens10M dataset
## Comparison with other solvers

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Passes</th>
<th>NN-PA-I</th>
<th>SGD</th>
<th>CD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Error Time</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Movielens10M</td>
<td>1</td>
<td>23.75 ± 0.05</td>
<td>31.58 ± 1.91</td>
<td>34.59 ± 0.03</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>20.91 ± 0.04</td>
<td>25.27 ± 0.02</td>
<td>21.38 ± 0.05</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>20.61 ± 0.01</td>
<td>24.54 ± 0.02</td>
<td>20.47 ± 0.01</td>
</tr>
<tr>
<td>Netflix</td>
<td>1</td>
<td>22.32 ± 0.01</td>
<td>27.29 ± 0.81</td>
<td>34.31 ± 0.01</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>20.01 ± 0.01</td>
<td>24.28 ± 0.01</td>
<td>21.60 ± 0.01</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>19.64 ± 0.01</td>
<td>23.70 ± 0.14</td>
<td>19.37 ± 0.01</td>
</tr>
<tr>
<td>Yahoo-Music</td>
<td>1</td>
<td>50.64 ± 0.33</td>
<td>52.52 ± 0.68</td>
<td>57.08 ± 0.28</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>38.44 ± 0.16</td>
<td>44.63 ± 1.24</td>
<td>45.32 ± 0.23</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>36.26 ± 0.09</td>
<td>41.62 ± 1.15</td>
<td>37.97 ± 0.21</td>
</tr>
</tbody>
</table>


Learned topic model

<table>
<thead>
<tr>
<th>Topic 1</th>
<th>Topic 2</th>
<th>Topic 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scream (Comedy, Horror, Thriller)</td>
<td>Dumb &amp; Dumber (Comedy)</td>
<td>Pocahontas (Animation, Children, Musical, ...)</td>
</tr>
<tr>
<td>The Fugitive (Thriller)</td>
<td>Ace Ventura: Pet Detective (Comedy)</td>
<td>Aladdin (Adventure, Animation, Children, ...)</td>
</tr>
<tr>
<td>The Blair Witch Project (Horror, Thriller)</td>
<td>Five Corners (Drama)</td>
<td>Merry Christmas Mr. Lawrence (Drama, War)</td>
</tr>
<tr>
<td>Deep Cover (Action, Crime, Thriller)</td>
<td>Ace Ventura: When Nature Calls (Comedy)</td>
<td>Toy Story (Adventure, Animation, Children, ...)</td>
</tr>
<tr>
<td>The Plague of the Zombies (Horror)</td>
<td>Jump Tomorrow (Comedy, Drama, Romance)</td>
<td>The Sword in the Stone (Animation, Children, Fantasy, ...)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Topic 4</th>
<th>Topic 5</th>
<th>Topic 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Belle de jour (Drama)</td>
<td>Four Weddings and a Funeral (Comedy, Romance)</td>
<td>Terminator 2: Judgment Day (Action, Sci-Fi)</td>
</tr>
<tr>
<td>Jack the Bear (Comedy, Drama)</td>
<td>The Birdcage (Comedy)</td>
<td>Braveheart (Action, Drama, War)</td>
</tr>
<tr>
<td>The Cabinet of Dr. Caligari (Crime, Drama, Fantasy, ...)</td>
<td>Shakespeare in Love (Comedy, Drama, Romance)</td>
<td>Aliens (Action, Horror, Sci-Fi)</td>
</tr>
<tr>
<td>M<em>A</em>S*H (Comedy, Drama, War)</td>
<td>Henri V (Drama, War)</td>
<td>Mortal Kombat (Action, Adventure, Fantasy)</td>
</tr>
<tr>
<td>Bed of Roses (Drama, Romance)</td>
<td>Three Men and a Baby (Comedy)</td>
<td>Congo (Action, Adventure, Mystery, ...)</td>
</tr>
</tbody>
</table>

6 out of 20 topics extracted from the Movielens10M dataset
Conclusion

- NMF is a widely-used method in machine learning and signal processing
- Its main applications are high-level feature extraction, denoising and matrix completion
- We proposed online passive-aggressive algorithms for the setting with missing values
References


References

Daniel D. Lee and H. Sebastian Seung.
Learning the parts of objects by non-negative matrix factorization.

Daniel D. Lee and H. Sebastian Seung.
Algorithms for non-negative matrix factorization.

Chih-Jen Lin.
Projected gradient methods for nonnegative matrix factorization.

Hsiang-Fu Yu, Cho-Jui Hsieh, Si Si, and Inderjit S. Dhillon.
Scalable coordinate descent approaches to parallel matrix factorization for recommender systems.