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

$$R \approx PQ$$

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

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

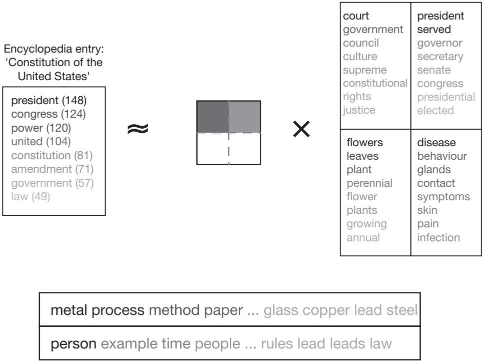
$$[r_{u,1}, \cdots, r_{u,d}] \approx \sum_{k=1}^{m} \underbrace{p_{u,k}}_{\text{weight / activation}} \times \underbrace{[q_{k,1}, \cdots, q_{k,d}]}_{\text{factor / basis / atom}}$$

- 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 $[r_{u,1}, \cdots, r_{u,d}]$ 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

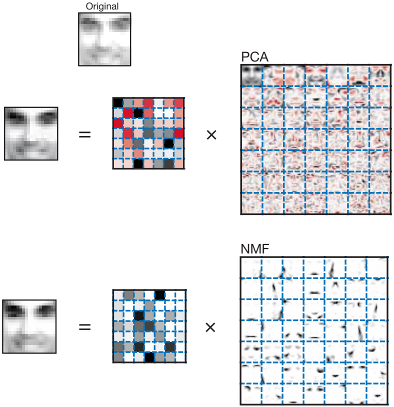


Using $n = 30,991$ articles from Grolier encyclopedia, vocabulary size $d = 15,276$ and number of topics $m = 200$ **[Lee & Seung, 99]**

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 $[q_{k,1}, \ldots, q_{k,d}]$ in Q correspond to image "parts"
- $p_{u,k}$ is the weight of part k in image u
  $\Rightarrow$ 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

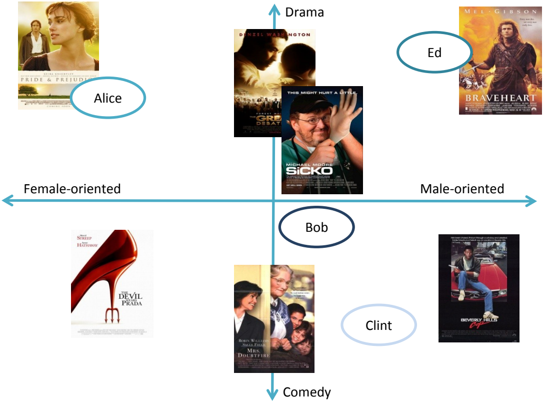
- R is a **partially observed** rating matrix

	Avatar	Escape From Alcatraz	K-Pax	Shawshank Redemption	Usual Suspects
Alice	1	2	?	5	4
Bob	3	?	2	5	3
Clint	?	?	?	?	2
Dave	5	?	4	4	5
Ethan	4	?	1	1	?

$r_{u,i}$

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

$$r_{u,i} \approx \underbrace{[p_{u,1}, \ldots, p_{u,m}]}_{\boldsymbol{p}_u} \underbrace{\begin{bmatrix} q_{1,i} \\ \vdots \\ q_{m,i} \end{bmatrix}}_{\boldsymbol{q}_i}$$

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:

$$\underset{P \geq 0, Q \geq 0}{\text{minimize}}\ F(P, Q) = \underbrace{\frac{1}{2}\|R - PQ\|^2}_{\text{error term}} + \underbrace{\frac{\lambda}{2}\left(\|P\|^2 + \|Q\|^2\right)}_{\text{regularization term}}$$

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:

$$\underset{P\geq 0, Q\geq 0}{\text{minimize}}\ F(P,Q) = \underbrace{D_I(R||PQ)}_{\text{error term}} + \underbrace{\frac{\lambda}{2}\left(\|P\|^2 + \|Q\|^2\right)}_{\text{regularization term}}$$

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

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**]

$$P_{u,k} \leftarrow P_{u,k} \frac{(RQ^{\mathrm{T}})_{u,k}}{(PQQ^T)_{u,k}} \quad Q_{k,i} \leftarrow Q_{k,i} \frac{(P^{\mathrm{T}}R)_{k,i}}{(P^T PQ)_{k,i}}$$

- 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**]

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

- η 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:

$$\underset{P \geq 0, Q \geq 0}{\text{minimize}} \; F(P, Q) = \frac{1}{2|\Omega|} \sum_{(u,i) \in \Omega} (\boldsymbol{r}_{u,i} - \boldsymbol{p}_u \cdot \boldsymbol{q}_i)^2 + \frac{\lambda}{2}\left(\|P\|^2 + \|Q\|^2\right)$$

where Ω is the set of observed values

Projected stochastic gradient method

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

$$\boldsymbol{p}_u \leftarrow \max\left(\boldsymbol{p}_u - \eta \nabla_P^{(u,k)} F(P, Q), 0\right)$$

$$\boldsymbol{q}_i \leftarrow \max\left(\boldsymbol{q}_i - \eta \nabla_Q^{(k,i)} F(P, Q), 0\right)$$

- 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 η ☹

Coordinate descent

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

$$P_{u,k} \leftarrow P_{u,k} + \operatorname*{argmin}_{\delta} F(P + E_{u,k}\delta, Q) \quad \text{or}$$
$$Q_{k,i} \leftarrow Q_{k,i} + \operatorname*{argmin}_{\delta} F(P, Q + E_{k,i}\delta)$$

  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 ☺

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

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

$$\begin{bmatrix} p_{1,1} & p_{1,2} & p_{1,3} \\ p_{2,1} & p_{2,2} & p_{2,3} \\ p_{3,1} & p_{3,2} & p_{3,3} \end{bmatrix} \quad \begin{bmatrix} ? & ? & ? \\ ? & ? & ? \\ ? & ? & ? \end{bmatrix}$$

Online algorithms

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

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

$$\begin{bmatrix} p_{1,1} & p_{1,2} & p_{1,3} \\ p_{2,1} & p_{2,2} & p_{2,3} \\ p_{3,1} & p_{3,2} & p_{3,3} \end{bmatrix} \quad \begin{bmatrix} ? & ? & ? \\ ? & ? & \color{red}{3} \\ ? & ? & ? \end{bmatrix}$$

Online algorithms

3. Update corresponding row of P and column of Q

$$\begin{bmatrix} q_{1,1} & q_{1,2} & \color{red}{q_{1,3}} \\ q_{2,1} & q_{2,2} & \color{red}{q_{2,3}} \\ q_{3,1} & q_{3,2} & \color{red}{q_{3,3}} \end{bmatrix}$$

$$\begin{bmatrix} p_{1,1} & p_{1,2} & p_{1,3} \\ \color{red}{p_{2,1}} & \color{red}{p_{2,2}} & \color{red}{p_{2,3}} \\ p_{3,1} & p_{3,2} & p_{3,3} \end{bmatrix} \quad \begin{bmatrix} ? & ? & ? \\ ? & ? & \color{red}{3} \\ ? & ? & ? \end{bmatrix}$$

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 $\boldsymbol{p}_{u_t}$ and $\boldsymbol{q}_{i_t}$ by

$$\boldsymbol{p}_{u_t}^{t+1} = \underset{\boldsymbol{p} \in \mathbf{R}_+^m}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{p} - \boldsymbol{p}_{u_t}^t\|^2 \text{ s.t. } |\boldsymbol{p} \cdot \boldsymbol{q}_{i_t}^t - r_{u_t,i_t}| = 0$$

$$\boldsymbol{q}_{i_t}^{t+1} = \underset{\boldsymbol{q} \in \mathbf{R}_+^m}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{q} - \boldsymbol{q}_{i_t}^t\|^2 \text{ s.t. } |\boldsymbol{p}_{u_t}^t \cdot \boldsymbol{q} - r_{u_t,i_t}| = 0$$

- 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

$$\begin{aligned}
\boldsymbol{w} &= \boldsymbol{p} && \text{or} \quad \boldsymbol{q} && \text{(variable)} \\
\boldsymbol{w}_{t+1} &= \boldsymbol{p}_{u_t}^{t+1} && \text{or} \quad \boldsymbol{q}_{i_t}^{t+1} && \text{(solution)} \\
\boldsymbol{w}_t &= \boldsymbol{p}_{u_t}^{t} && \text{or} \quad \boldsymbol{q}_{i_t}^{t} && \text{(current iterate)} \\
\boldsymbol{x}_t &= \boldsymbol{q}_{i_t}^{t} && \text{or} \quad \boldsymbol{p}_{u_t}^{t} && \text{(input)} \\
y_t &= r_{u_t,i_t} && && \text{(target)}
\end{aligned}$$

Passive-aggressive algorithms for NMF

- Allow to not perfectly fit the target

$$\boldsymbol{w}_{t+1} = \underset{\boldsymbol{w} \in \mathbf{R}_{+}^{m}}{\operatorname{argmin}} \frac{1}{2} \|\boldsymbol{w} - \boldsymbol{w}_t\|^2 \text{ s.t. } |\boldsymbol{w} \cdot \boldsymbol{x}_t - y_t| \leq \epsilon$$

- If $|\boldsymbol{w}_t \cdot \boldsymbol{x}_t - y_t| \leq \epsilon$, the algorithm is "passive", i.e., $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t$
- 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 $\Rightarrow$ potential overfitting
- Introduce a slack variable to allow some error

$$\boldsymbol{w}_{t+1},\ \xi^* = \underset{\boldsymbol{w}\in\mathbf{R}_+^m,\ \xi\in\mathbf{R}_+}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{w}-\boldsymbol{w}_t\|^2 + C\xi$$

$$\text{s.t. } |\boldsymbol{w}\cdot\boldsymbol{x}_t - y_t| \leq \epsilon + \xi,$$

- $C > 0$ controls the trade-off between being conservative and corrective

Passive-aggressive algorithms for NMF

- The solution is of the form

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

  where κ and θ are non-negative scalars

- In our AISTATS paper, we present three $O(m)$ methods for finding κ and θ [**Blondel et al., 2014**]
  - An exact method
  - A bisection method
  - An approximate update method

Passive-aggressive algorithms for NMF

Difference between the effect of ϵ and C

- Increasing ϵ 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

Solver	Iteration cost	Online	Hyper-parameter
Multiplicative	☹	☹	☺
Projected grad.	☹	☹	☺
Projected stochastic grad.	☺	☺	☹
Coordinate descent	☺	☹	☺
Passive-Aggressive	☺	☺	☺

In the setting with missing values.

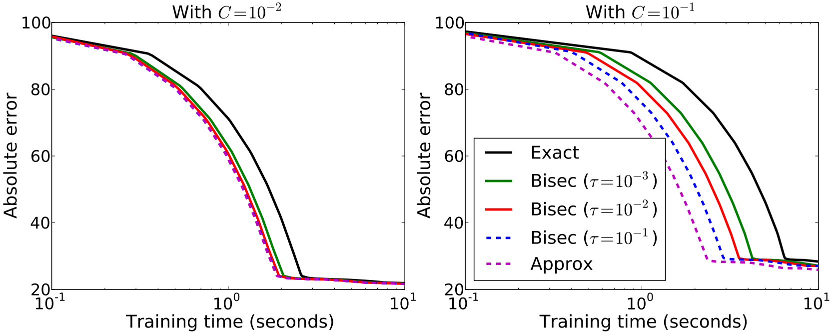
Experimental results

- Datasets used

Dataset	Users	Items	Ratings
Movielens10M	69,878	10,677	10,000,054
Netflix	480,189	17,770	100,480,507
Yahoo-Music	1,000,990	624,961	252,800,275

- 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

Dataset	Passes		NN-PA-I	SGD	CD
Movielens10M	1	Error	**23.75 ± 0.05**	31.58 ± 1.91	34.59 ± 0.03
		Time	3.24 ± 0.01	2.68 ± 0.01	3.88 ± 0.01
	3	Error	**20.91 ± 0.04**	25.27 ± 0.02	21.38 ± 0.05
		Time	10.28 ± 0.01	8.09 ± 0.08	12.73 ± 0.01
	5	Error	20.61 ± 0.01	24.54 ± 0.02	**20.47 ± 0.01**
		Time	17.40 ± 0.06	13.44 ± 0.03	22.57 ± 0.01
Netflix	1	Error	**22.32 ± 0.01**	27.29 ± 0.81	34.31 ± 0.01
		Time	34.29 ± 0.10	27.68 ± 0.41	36.58 ± 0.37
	3	Error	**20.01 ± 0.01**	24.28 ± 0.01	21.60 ± 0.01
		Time	109.53 ± 2.97	82.98 ± 0.14	153.46 ± 0.72
	5	Error	19.64 ± 0.01	23.70 ± 0.14	**19.37 ± 0.01**
		Time	181.43 ± 0.22	133.59 ± 0.60	270.28 ± 0.49
Yahoo-Music	1	Error	**50.64 ± 0.33**	52.52 ± 0.68	57.08 ± 0.28
		Time	114.16 ± 0.05	96.89 ± 0.04	170.38 ± 0.06
	3	Error	**38.44 ± 0.16**	44.63 ± 1.24	45.32 ± 0.23
		Time	335.13 ± 0.34	291.59 ± 0.24	468.86 ± 0.69
	5	Error	**36.26 ± 0.09**	41.62 ± 1.15	37.97 ± 0.21
		Time	576.08 ± 0.73	475.86 ± 2.90	787.57 ± 1.68

Learned topic model

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

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

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

Mathieu Blondel, Yotaro Kubo, and Naonori Ueda.
Online Passive-Aggressive Algorithms for Non-Negative Matrix Factorization and Completion
Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistic, 96–104, 2014.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer.
Online passive-aggressive algorithms.
Journal of Machine Learning Research, 7:551–585, 2006.

Cho-Jui Hsieh and Inderjit S. Dhillon.
Fast coordinate descent methods with variable selection for non-negative matrix factorization.
In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1064–1072, 2011.

Cédric Févotte, A Taylan Cemgil.
Nonnegative matrix factorizations as probabilistic inference in composite models.
In *Proceedings of EUSIPCO*, pages 1913–1917, 2009.

References

Daniel D. Lee and H. Sebastian Seung.
Learning the parts of objects by non-negative matrix factorization.
Nature, 401(6755):788–791, 1999.

Daniel D. Lee and H. Sebastian Seung.
Algorithms for non-negative matrix factorization.
In *Advances in Neural Information Processing Systems 13*, pages 556–562, 2001.

Chih-Jen Lin.
Projected gradient methods for nonnegative matrix factorization.
Neural Computation, 19(10):2756–2779, 2007.

Hsiang-Fu Yu, Cho-Jui Hsieh, Si Si, and Inderjit S. Dhillon.
Scalable coordinate descent approaches to parallel matrix factorization for recommender systems.
In *ICDM*, pages 765–774, 2012.