Exercises
Computational Intelligence Lab
SS 2019

Machine Learning Institute
Dept. of Computer Science, ETH Zürich
Prof. Dr. Thomas Hofmann
Web http://cil.inf.ethz.ch/

Series 4, March 11, 2019 (Matrix Approximation & Reconstruction)

Problem 1 (Alternating Least Squares for Collaborative Filtering):

Suppose we have a user-movie rating matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ which contains the ratings from m users for n movies on Netflix. Let $\mathcal{I}$ contain the indexes of the entries observed in $\mathbf{A}$.

To build a recommender system, we decompose the rating matrix $\mathbf{A}$ into a product of two matrices $\mathbf{UV}$, where $\mathbf{U} \in \mathbb{R}^{m\times k}$ and $\mathbf{V} \in \mathbb{R}^{k\times n}$ are the user and item matrices correspondingly, and the number of factors $k \ll \max(n, m)$ is relatively small. Let the i-th row of $\mathbf{U}$ be $\mathbf{u}_i^\top$ ($\mathbf{u}_i \in \mathbb{R}^k$), and the j-th column of $\mathbf{V}$ be $\mathbf{v}_j \in \mathbb{R}^k$. Let $\mathbf{I}_k$ denote the $k \times k$ identity matrix.

We are going to perform approximate matrix factorization by minimizing the regularized Frobenius loss defined as follows:

$$L(\mathbf{U},\mathbf{V}) = \sum_{(i,j)\in\mathcal{I}} (a_{ij} - \mathbf{u}_i^\top \mathbf{v}_j)^2 + \lambda \sum_{i=1}^{m} \|\mathbf{u}_i\|^2 + \lambda \sum_{j=1}^{n} \|\mathbf{v}_j\|^2, \tag{1}$$

where $\lambda > 0$ is the regularization strength.

The Alternating Least Squares algorithm minimizes the loss function $L(\mathbf{U},\mathbf{V})$ in the following way.

Algorithm 1: Alternating Least Squares (ALS)

```
Initialize $\mathbf{U}, \mathbf{V}$
while not convergent do
    for $i = 1, \ldots, m$ do
        $\mathbf{u}_i = (\sum_{j:(i,j)\in\mathcal{I}} \mathbf{v}_j\mathbf{v}_j^\top + \lambda\mathbf{I}_k)^{-1} \sum_{j:(i,j)\in\mathcal{I}} a_{ij}\mathbf{v}_j$
    for $j = 1, \ldots, n$ do
        $\mathbf{v}_j = (\sum_{i:(i,j)\in\mathcal{I}} \mathbf{u}_i\mathbf{u}_i^\top + \lambda\mathbf{I}_k)^{-1} \sum_{i:(i,j)\in\mathcal{I}} a_{ij}\mathbf{u}_i$
```

1. Is the objective function (1) convex with respect to the pair $(\mathbf{U},\mathbf{V})$? If not, prove it.
2. Is the objective (1) convex with respect to $\mathbf{U}$?
3. Derive the update rule for $\mathbf{u}_i$. Note that the update rule for $\mathbf{v}_j$ is symmetric to that for $\mathbf{u}_i$.
   Hint: differentiate the objective (1) with respect to $\mathbf{u}_i$ holding $\mathbf{V}$ constant and set the gradient to zero.
4. Suppose the computational complexity of inverting a $k \times k$ matrix is $O(k^3)$, let n_i be the number of items rated by user i. Find the computational complexity of the step $\mathbf{u}_i = (\sum_{j:(i,j)\in\mathcal{I}} \mathbf{v}_j\mathbf{v}_j^\top + \lambda\mathbf{I}_k)^{-1} \sum_{j:(i,j)\in\mathcal{I}} a_{ij}\mathbf{v}_j$ in the ALS algorithm above. Use big O notation.
5. For a recommender system, $\mathbf{u}_i$ and $\mathbf{v}_j$ can be interpreted as the low-dimensional representations of user i and item j correspondingly. Interpret the update steps of the ALS algorithm in terms of obtaining low-dimensional representations for a recommender system.

Problem 2 (Convex Relaxation for Exact Matrix Recovery):

In the lecture, we relaxed the NP-hard problem of Low-Rank Matrix Reconstruction to a convex optimization problem by replacing the rank objective with the nuclear norm objective (Exact Matrix Reconstruction problem) and considered solving it via SVD Shrinkage Iterations.

1. We start with matrix norms and proving the nuclear norm $\|\mathbf{B}\|_*$ of a matrix $\mathbf{B} \in \mathbb{R}^{m\times n}$ from the unit Euclidean ball ($\|\mathbf{B}\|_2 \leq 1$) to be a lower bound of rank($\mathbf{B}$). As we know (derive if it does not seem obvious), the Frobenius norm of a matrix is equal to the square root of the sum of its squared singular values, whereas the nuclear norm (the Schatten 1-norm) of a matrix is equal to the sum of its singular values,

$$\|\mathbf{A}\|_F = \|\sigma(\mathbf{A})\|_2, \quad \|\mathbf{A}\|_* = \|\sigma(\mathbf{A})\|_1. \tag{2}$$

Analogously, the Euclidean operator norm of a matrix can be defined as its largest singular value:

$$\|\mathbf{A}\|_2 = \sigma_{\max}(\mathbf{A}) = \|\sigma(\mathbf{A})\|_\infty. \tag{3}$$

Prove the following inequality from lecture,

$$\text{rank}(\mathbf{A}) \geq \|\mathbf{A}\|_* \text{ for } \|\mathbf{A}\|_2 \leq 1. \tag{4}$$

2. Prove that the nuclear norm is a convex function (in fact, every norm function is convex) to confirm that the Exact Matrix Reconstruction problem

$$\min_{\mathbf{B}} \|\mathbf{B}\|_*, \quad \text{s.t. } \|\mathbf{A} - \mathbf{B}\|_{\mathbf{G}} = 0, \tag{5}$$

is indeed a relaxation of the non-convex problem of Low-Rank Matrix Reconstruction

$$\min_{\mathbf{B}} \text{rank}(\mathbf{B}), \quad \text{s.t. } \|\mathbf{A} - \mathbf{B}\|_{\mathbf{G}} = 0, \tag{6}$$

where $\mathbf{G}$ is a binary matrix of partial observations (refer to the lecture slides for notation).

3. Even though solving relaxation (5) via SVD Shrinkage Iterations should be considered as preferential in practice, to consolidate our understanding, let us now consider another approach, using the framework of positive semidefinite programming.

Reformulate the relaxation (5) as a problem of semidefinite programming (SDP) in the following form,

$$\min_{\mathbf{B},\mathbf{W_1},\mathbf{W_2}} \tfrac{1}{2}\text{trace}(\mathbf{W_1}) + \tfrac{1}{2}\text{trace}(\mathbf{W_2}), \quad \text{s.t. } \underbrace{\begin{bmatrix} \mathbf{W_1} & \mathbf{B} \\ \mathbf{B}^\intercal & \mathbf{W_2} \end{bmatrix} \succeq 0,}_{\text{cone constraints}} \; \underbrace{\|\mathbf{A} - \mathbf{B}\|_{\mathbf{G}} = 0}_{\text{affine constraints}}. \tag{7}$$

There have been multiple solvers developed for SDP problems that can be used directly to solve the relaxation (7). However, for a large number of parameters (large matrix $\mathbf{A}$), the computational complexity of optimization algorithms involved is often very high and this approach is rarely used in practice.

[1] Benjamin Recht, Maryam Fazel, and Pablo A. Parrilo. *Guaranteed Minimum-Rank Solutions of Linear Matrix Equations via Nuclear Norm Minimization*. SIAM Review 2010 52:3, 471-501. `https://arxiv.org/pdf/0706.4138.pdf`

Problem 3 (Stochastic Gradient Descent for Collaborative Filtering):

We have seen matrix completion already in Exercise 2, where we approximated a full matrix by an SVD.

In this exercise, we will apply *optimization techniques* to directly minimize the training error for the (unconstrained) matrix factorization formulation $\min_{\mathbf{U}\in Q_1,\mathbf{Z}\in Q_2} f(\mathbf{U},\mathbf{Z})$, with the objective function being the mean squared error

$$f(\mathbf{U},\mathbf{Z}) = \frac{1}{|\Omega|} \sum_{(d,n)\in\Omega} \frac{1}{2}\left[\mathbf{X}_{dn} - (\mathbf{U}\mathbf{Z}^T)_{dn}\right]^2 \tag{8}$$

and $\mathbf{U} \in Q_1 := \mathbb{R}^{D\times K}$, $\mathbf{Z} \in Q_2 := \mathbb{R}^{N\times K}$.

Here $\Omega \subseteq [D] \times [N]$ is the set of the indices of the observed ratings of the input matrix $\mathbf{X}$.

Environment setup:

Please use the same setup and data as in Exercise 2, as also explained on the web page for the collaborative filtering project task:

https://www.kaggle.com/c/cil-collab-filtering-2019.

Task: Implement Stochastic Gradient Descent

1. Derive the full gradient $\nabla_{(\mathbf{U},\mathbf{Z})} f(\mathbf{U},\mathbf{Z})$. Note that since we have $(D+N) \times K$ variables, the gradient here can be seen as a $(D+N) \times K$ matrix.
2. Derive a stochastic gradient $\mathbf{G}$ using the sum structure of f over the Ω elements. We want to do this in such a way that $\mathbf{G}$ only depends on a single observed rating $(d,n) \in \Omega$.
3. Implement the Stochastic Gradient Descent algorithm[1], for our objective function given in (8).
4. Experimentally find the best stepsize γ to obtain the lowest training error value.
5. Does the test error also decrease monotonically during optimization, or does it increase again after some time?
6. (OPTIONAL: Can you speed up your code, by for example maintaining the set of values $(\mathbf{U}\mathbf{Z}^\top)_{dn}$ for the few observed values $(d,n) \in \Omega$, and thereby avoiding the computation of the matrix multiplication $\mathbf{U}\mathbf{Z}^\top$ in every step?)

Extensions: Naturally there are many ways to improve your solution. One of them is to use regularization term to avoid over-fitting. Such techniques and other extensions can be found e.g. in the following publications:

[1] Webb, B. (2006). *Netflix Update: Try This at Home*. Simon Funk's Personal Blog. `http://sifter.org/~simon/journal/20061211.html`

[2] Koren Y., Bell R., Volinsky B. (2009). *Matrix Factorization Techniques for Recommender Systems*. IEEE Computer, Volume 42, Issue 8, pp. 30-37. `http://research.yahoo.com/files/ieeecomputer.pdf`

[3] A. Paterek (2007). *Improving Regularized Singular Value Decomposition for Collaborative Filtering*. Proc. KDD Cup and Workshop, ACM Press, pp. 39-42.

[1] `https://en.wikipedia.org/wiki/Stochastic_gradient_descent`