Exercises
Computational Intelligence Lab
SS 2020

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

Series 5, March 16, 2020 (Non-Negative Matrix Factorization)

Problem 1 (Convex Relaxation for Exact Matrix Recovery):

In the past lecture, we relaxed the NP-hard problem of *Exact Matrix Reconstruction* to a convex optimization problem by replacing the rank objective with the nuclear norm objective 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,
$$\|\mathbf{A}\|_F = \|\sigma(\mathbf{A})\|_2, \quad \|\mathbf{A}\|_* = \|\sigma(\mathbf{A})\|_1.$$
Analogously, the Euclidean operator norm of a matrix can be defined as its largest singular value:
$$\|\mathbf{A}\|_2 = \sigma_{\max}(\mathbf{A}) = \|\sigma(\mathbf{A})\|_\infty.$$
Prove the following inequality from the lecture,
$$\text{rank}(\mathbf{A}) \geq \|\mathbf{A}\|_* \text{ for } \|\mathbf{A}\|_2 \leq 1.$$

2. Prove that the nuclear norm is a convex function (in fact, every norm function is convex) to confirm that the problem
$$\min_{\mathbf{B}} \|\mathbf{B}\|_*, \quad \text{s.t. } \|\mathbf{A}-\mathbf{B}\|_{\mathbf{G}} = 0, \tag{1}$$
is indeed a relaxation of the non-convex Exact Matrix Reconstruction problem,
$$\min_{\mathbf{B}} \text{rank}(\mathbf{B}), \quad \text{s.t. } \|\mathbf{A}-\mathbf{B}\|_{\mathbf{G}} = 0, \tag{2}$$
where $\mathbf{G}$ is a binary matrix of partial observations (refer to the lecture slides for notation).

3. Even though solving relaxation (1) 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 semi-definite programming.
Reformulate the relaxation (1) as a problem of semi-definite programming (SDP) in the following form,
$$\min_{\mathbf{B},\mathbf{W_1},\mathbf{W_2}} \quad \frac{1}{2}\text{Tr}(\mathbf{W_1}) + \frac{1}{2}\text{Tr}(\mathbf{W_2}) \tag{3}$$
$$\text{subject to } \underbrace{\begin{bmatrix}\mathbf{W_1} & \mathbf{B}\\ \mathbf{B}^\mathsf{T} & \mathbf{W_2}\end{bmatrix} \succeq 0}_{\text{cone constraints}} \text{ and } \underbrace{\|\mathbf{A}-\mathbf{B}\|_{\mathbf{G}} = 0}_{\text{affine constraints}}. \tag{4}$$
There have been multiple solvers developed for SDP problems that can be used directly to solve the relaxation (3). 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 2 (pLSA and LDA theory, 1):

Get familiar with the two methods for information retrieval discussed in the lecture: pLSA and LDA. Carefully go through the reasoning and the final formulas — will be important for the exam to understand such derivations. If you feel like details are missing, or you are interested in a more detailed discussion, please check the original papers:

https://arxiv.org/pdf/1301.6705.pdf

http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

Problem 3 (pLSA and LDA theory, 2):

Under the notation introduced in the lecture (slide 10), the log-likelihood cost for pLSA is

$$\begin{aligned}\log \ell(\boldsymbol{U}, \boldsymbol{V}) &= \sum_{ij} x_{ij} \log \sum_{z} p(w = j|z)p(z|d = i) \\ &= \sum_{ij} x_{ij} \log \sum_{z} u_{zi} v_{zj}\end{aligned}$$

i. We want to solve

$$\max_{\boldsymbol{U},\boldsymbol{V}} \ \log \ell(\boldsymbol{U}, \boldsymbol{V})$$

$$\text{subject to } \sum_{z} u_{zi} = 1, \ \sum_{j} v_{zj} = 1, \ u_{zi} \geq 0, \ v_{zj} \geq 0.$$

Is this problem convex? Can the solution be computed in closed-form?

ii. Suppose that the latent variables z were observed (we know, for each word in each document, the topic (color) that "generated" it). In this case, can we compute the solution in a closed form?

Bonus: If self-isolation for Covid-19 got you bored, try to solve points (iii) and (iv) from last year sheet at http://www.da.inf.ethz.ch/teaching/2019/CIL/exercises/solution05.pdf

Problem 4 (Implementing pLSA for Discovering Topics in a Corpus):

In this question, we are going to use pLSA to discover topics in a corpus of documents. We will use a preprocessed dataset of documents from the Associated Press (courtesy of https://github.com/kzhai/PyLDA) which contains a collection of 2221 documents, "doc.dat," which has been conveniently split into train and test sets of sizes 2000 and 221 documents respectively.

Fortunately, thanks to the pre-processed data and some handy libraries, you do not need to do any additional data wrangling. Everything is ready for training models.

Setup:

- Download the Associated Press corpus, "associated-press.tar.gz" and the Jupyter notebook template "pLSA-for-the-AP.ipynb" from the lecture's github repository (link below).
- Install the dependencies listed in the "README.md" file.
- Follow instructions in the notebook. You will implement the Expectation-Maximization algorithm and do some basic analysis of the resulting model and learned topics.

https://github.com/dalab/lecture_cil_public/tree/master/exercises/ex5

Questions:

1. Why does the lower bound increase on each iteration?
2. Why does the log-likelihood increase on each iteration?
3. What are the learned topics?
4. What is the best choice for the number of topics? How do you know?