Embarrassingly Shallow Autoencoders for Sparse Data

Author	Harald Steck
Created	12.27.2020
Journal	Proceedings of The 2019 Web Conference (WWW)
Pages	7
Reviewed	☐

Introduction

심플한 모델을 사용하게 된 계기

Unlike in areas like computer vision, it was found that a small number of hidden layers achieved the best recommendation accuracy. → We take this to the extreme, and define a **linear model without a hidden layer. The binary input vector indicates which items a user has interacted with, and the model's objective in its output layer is to predict the best items to recommend to the user. This is done by reproducing the input as its output, as is typical for autoencoders.**

Model

▼ Notations

U = len(users), I = len(items)

X: sparse, binary matrix (implicit feedback data) with size of $U * I$

B: item-item weight matrix with size of $I * I$

$diag(B) = 0$: **diagonal of the weight matrix is constrained to 0**

$S_{uj} = X_u * B_j$: predicted score for an item j given to user u

Training

Convex objective for learning weights B: $min_B ||X - XB||_F^2 + \lambda * ||B||_F^2$
s.t. $diag(B) = 0$

$||\ \ ||_F$ denotes the Frobenius norm (= matrix norm) between X and the predicted scores $S = XB$ over other loss functions because it allows for a **closed-form solution**.

L2-norm regularization is used when learning the weights B. Thus, our hyper-parameter is λ.

Closed-Form Solution

1. Include the equality constraint ($diag(B) = 0$) into the objective function by forming the Lagrangian. $\gamma = (\gamma_1, ..., \gamma_{|I|})^T$ is the vector of the Lagrangian multipliers.

$$L = ||X - XB||_F^2 + \lambda * ||B||_F^2 + 2 * \gamma^T * diag(B)$$

2. The constrained optimization problem above is solved by minimizing this Lagrangian. As a necessary condition, we set its derivative to zero, which yields:

$$\hat{B} = (X^T X + \lambda I)^{-1} * (X^T X - diagMat(\lambda))$$

   $diagMat(\ )$ denotes the diagonal matrix and I the identity matrix.

   Define the inverse part as: $\hat{P} = (X^T X + \lambda I)^{-1}$

3. Then, the equation from 2. becomes:

$$
\begin{aligned}
\hat{B} &= (X^\top X + \lambda I)^{-1} \cdot (X^\top X - \text{diagMat}(\gamma)) \\
&= \hat{P} \cdot (\hat{P}^{-1} - \lambda I - \text{diagMat}(\gamma)) \\
&= I - \hat{P} \cdot (\lambda I + \text{diagMat}(\gamma)) \\
&= I - \hat{P} \cdot \text{diagMat}(\tilde{\gamma})
\end{aligned}
$$

So, eventually, the learned weights are given by:

$$
\hat{B}_{i,j} = \begin{cases} 0 & \text{if } \ i = j \\ -\frac{\hat{P}_{ij}}{\hat{P}_{jj}} & \text{otherwise.} \end{cases}
$$

Algorithm

The training of EASE only requires the item-item matrix $G = X^T X$ as input, and hence is efficient if the size of G is smaller than the number of user-item interactions in X.

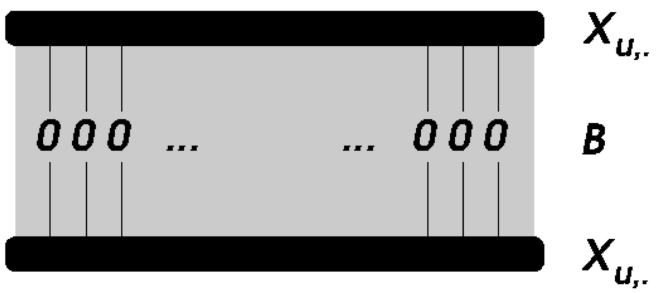
This also means that EASE will be fast and efficient when there are a lot of users and less items, but **when there are a lot of items**, the training process may be computationally expensive.

Algorithm 1: Training in Python 2 using numpy

Input: data Gram-matrix $G := X^\top X \in \mathbb{R}^{|\mathcal{I}|\times|\mathcal{I}|}$,
L2-norm regularization-parameter $\lambda \in \mathbb{R}^+$.
Output: weight-matrix B with zero diagonal (see Eq. 8).
$diagIndices$ = numpy.diag_indices(G.shape[0])
$G[diagIndices]$ += λ
P = numpy.linalg.inv(G)
B = P / (-numpy.diag(P))
$B[diagIndices]$ = 0

Important Points

- The **self-similarity of each item is constrained to 0** between the input & output layers.



- The convex training objective derives the closed-form solution of EASE.
- Instead of using **any hidden layer**, the self-similarity constraint of each item in the input and output layer is used, and this may be more effective on sparse data!

Comparison with Neighborhood-based Approaches

The closed-form solution of EASE's convex training objective shows that the neighborhood-based approaches used in collaborative filtering are based on **conceptually incorrect item-item similarity matrices**, (neighborhood-based models usually use the co-occurrence matrix (= X'X) as an item-item or user-user similarity matrix) while EASE may be considered a principled neighborhood model.

In other words, the conceptually correct similarity matrix to be used in neighborhood approaches is based on the *inverse* of the given co-occurrence matrix.

Important Points

- Neighborhood-based models actually use the **incorrect item-item similarity matrices**.
- Without using complex and long codes with deep non-linear NNs, EASE outperforms other models as well as neighborhood-based approaches.

Difficulties

Although EASE is a simple and effective model which combines the strengths of auto-encoders and neighborhood-based approaches, it is computationally **very expensive** ($O(n^3)$) to calculate the inverse matrix of $X^T X$, especially when the size of the item set is very large.

For example, in the experiments of this paper, the datasets used were the following:

- MovieLens 20 Million (ML-20M) data: 136,677 users and **20,108** movies with about 10 million interactions
- Netflix Prize data: 463,435 users and **17,769** movies with about 57 million interactions
- Million Song Data (MSD): 571,355 users and **41,140** songs with about 34 million interactions

However, sometimes, (such as music recommendation, ex. Playlist Continuation) the item set can be much larger.

This may be the reason that in real-life applications, other methods such as VAEs (generative model), Graph-NNs, and sometimes even the usual neighborhood-based models are used in recommender systems.