

Generalizing Linear Autoencoder Recommenders with Decoupled Expected Quadratic Loss

Ruixin Guo^{1*†}, Xinyu Li^{1*}, Hao Zhou¹, Yang Zhou², Ruoming Jin¹

1. Department of Computer Science, Kent State University

2. Department of Computer Science and Software Engineering, Auburn University

* Equal contribution † Presenter

Rio de Janeiro, April 23-27, 2026



ICLR
International Conference On
Learning Representations

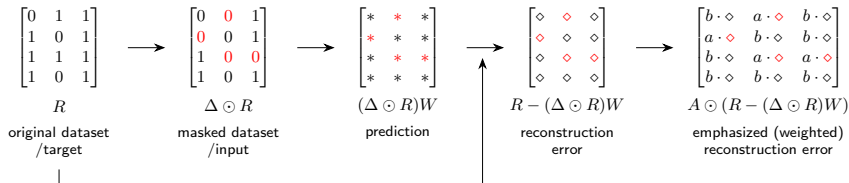
Introduction

In recent years, Linear Autoencoders (LAEs) have demonstrated surprisingly strong performance in recommender systems, even outperforming deep neural network models.

EDLAE [1] is one of the state-of-the-art LAE models, which obtains the optimal model W^* by minimizing the following reconstruction loss:

$$W^* = \arg \min_W \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N \|A^{(k)} \odot (R - (\Delta^{(k)} \odot R)W)\|_F^2 \quad (1)$$

- $R \in \{0, 1\}^{m \times n}$: the dataset, with m users and n items.
- $W \in \mathbb{R}^{n \times n}$: the model, a matrix representing item-item relationships.
- $\Delta^{(k)} \in \{0, 1\}^{m \times n}$: a random dropout matrix. Δ_{ij} i.i.d. drawn from $\text{Bern}(1 - p)$.
- $A^{(k)}$: an emphasizing matrix. $A_{ij}^{(k)} = a$ if $\Delta_{ij} = 0$, and $A_{ij}^{(k)} = b$ if $\Delta_{ij} = 1$.
Set $a \geq b \geq 0$ to put greater weight on the reconstruction loss of dropped entries.



Introduction

When $b = 0$, EDLAE has a closed-form solution for W^* :

$$W^* = \frac{1}{1-p} \left(I - C \cdot (I \odot C)^{-1} \right), \text{ where } C = \left(R^T R + \frac{p}{1-p} I \odot R^T R \right)^{-1} \quad (2)$$

In this case, the diagonal of W^* is zero.

The main issue is that the closed-form solution of EDLAE is not derived from a [statistical perspective](#), which makes closed-form analysis difficult. Under the original framework, it is hard to answer the following questions:

- Does solutions with $b = 0$ always yield the best test performance?
- When $b > 0$, do closed-form solutions exist? Do they perform better?
- Does emphasizing dropped entries (i.e., choosing $a \geq b \geq 0$) always lead to good performance? What if we instead de-emphasize them by choosing $b > a \geq 0$?
- Why must W^* have a zero-diagonal? Would allowing a non-zero diagonal improve performance?

To overcome this limitation, we propose a new framework from the [statistical perspective](#) to facilitate closed-form analysis of EDLAE.

Decoupled Expected Quadratic Loss

By the Law of Large Numbers, we can rewrite the EDLAE objective (1) in its expectation form as follows:

$$\begin{aligned} W^* &= \arg \min_W l_{\mathcal{B}}(W), \text{ where } l_{\mathcal{B}}(W) = \mathbb{E}_{\Delta \sim \mathcal{B}} [\|A \odot (R - (\Delta \odot R)W)\|_F^2] \\ &= \sum_{i=1}^n \mathbb{E}_{\Delta \sim \mathcal{B}} [\|A^{(i)} R_{*i} - A^{(i)} (\Delta \odot R) W_{*i}\|_F^2], \quad A^{(i)} = \text{diag}(A_{*i}) \end{aligned} \quad (3)$$

Let $X = A^{(i)} (\Delta \odot R)$ be the input and $Y_{*i} = A^{(i)} R_{*i}$ be the target, then both X and Y are **random variables**.

This statistical point of view – **considering input and target as random variables and taking expectation of the loss with respect to them** – inspires us to propose a **general** loss function, the **Decoupled Expected Quadratic Loss (DEQL)**:

$$l_{\mathcal{D}}(W) = \sum_{i=1}^n h_{\mathcal{D}^{(i)}}^i(W_{*i}), \text{ where } h_{\mathcal{D}^{(i)}}^i(W_{*i}) = \mathbb{E}_{(X,Y) \sim \mathcal{D}^{(i)}} [\|Y_{*i} - XW_{*i}\|_F^2] \quad (4)$$

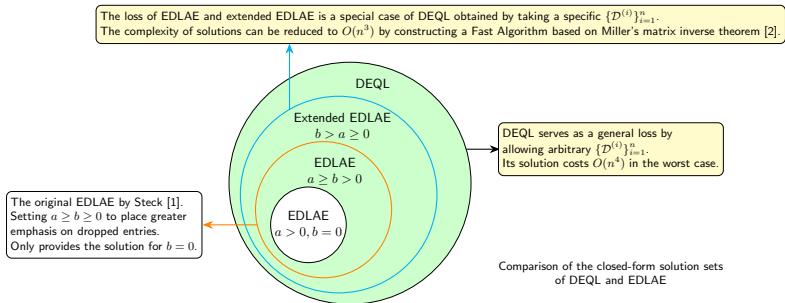
DEQL (4) holds for any collection of distributions $\{\mathcal{D}^{(i)}\}_{i=1}^n$. EDLAE (3) is a **special case** of (4) obtained by choosing a specific collection.

Decoupled Expected Quadratic Loss

Advantages of DEQL:

- It enables a simpler and better-understood closed-form analysis.
- It extends the solution space, allowing us to obtain solutions for $b > 0$ and overcoming the $b = 0$ limitation of EDLAE.
- Solutions with $b > 0$ can achieve better test performance than those with $b = 0$.
- It supports choosing arbitrary distributions $\{\mathcal{D}^{(i)}\}_{i=1}^n$, which could facilitate designing objectives that outperform EDLAE.

Overall Framework:



Closed-Form Solutions of DEQL: $b > 0$ Case

When $b > 0$, the optimal solution of DEQL W^* is solved column-wisely by

$$W_{*i}^* = \mathbb{E}_{(X,Y) \sim \mathcal{D}^{(i)}} \left[X^T X \right]^{-1} \mathbb{E}_{(X,Y) \sim \mathcal{D}^{(i)}} \left[X^T Y_{*i} \right] \text{ for } i = 1, 2, \dots, n$$

Applying to EDLAE case, we plug in $X = A^{(i)}(\Delta \odot R)$ and $Y_{*i} = A^{(i)}R_{*i}$ and get

$$W_{*i}^* = H^{(i)-1} v^{(i)} \text{ for } i = 1, 2, \dots, n, \text{ where} \quad (5)$$

$$H^{(i)} = \mathbb{E}_{\Delta \sim \mathcal{B}} \left[(\Delta \odot R)^T A^{(i)2} (\Delta \odot R) \right], \quad v^{(i)} = \mathbb{E}_{\Delta \sim \mathcal{B}} \left[(\Delta \odot R)^T A^{(i)2} R_{*i} \right]$$

where $H^{(i)}$ and $v^{(i)}$ can be expressed in a more explicit form

Lemma 1

$H^{(i)} = G^{(i)} \odot R^T R$ and $v^{(i)} = u^{(i)} \odot R^T R_{*i}$, where

$$G_{kl}^{(i)} = \begin{cases} (1-p)b^2 & \text{if } k=l=i \\ (1-p)^2b^2 & \text{if } k \neq l=i \text{ or } l \neq k=i \\ (1-p)pa^2 + (1-p)^2b^2 & \text{if } k=l \neq i \\ (1-p)^2pa^2 + (1-p)^3b^2 & \text{if } i \neq k \neq l \neq i \end{cases}, \quad u_k^{(i)} = \begin{cases} (1-p)b^2 & \text{if } k=i \\ (1-p)pa^2 + (1-p)^2b^2 & \text{if } k \neq i \end{cases}$$

for $k, l \in \{1, 2, \dots, n\}$.

A Fast Algorithm for Computing $b > 0$ Solutions

The major challenge in computing $b > 0$ solutions by $W_{*i}^* = H^{(i)-1}v^{(i)}$ for all i is **computational inefficiency**.

Note that $H^{(i)} = G^{(i)} \odot R^T R$, and each $G^{(i)}$ differs only in the i th row and column:

$$\begin{bmatrix} \times & * & * & * & * \\ * & + & \circ & \circ & \circ \\ * & \circ & + & \circ & \circ \\ * & \circ & \circ & + & \circ \\ * & \circ & \circ & \circ & + \end{bmatrix} \quad G^{(1)} \quad \begin{bmatrix} + & * & \circ & \circ & \circ \\ * & \times & * & * & * \\ \circ & * & + & \circ & \circ \\ \circ & * & \circ & + & \circ \\ \circ & * & \circ & \circ & + \end{bmatrix} \quad G^{(2)} \quad \dots \quad \begin{bmatrix} + & \circ & \circ & \circ & * \\ \circ & + & \circ & \circ & * \\ \circ & \circ & + & \circ & * \\ \circ & \circ & \circ & + & * \\ * & * & * & * & \times \end{bmatrix} \quad G^{(n)}$$

$$\times : (1-p)b^2, \quad * : (1-p)^2b^2, \quad + : (1-p)pa^2 + (1-p)^2b^2, \quad \circ : (1-p)^2pa^2 + (1-p)^3b^2$$

Each column W_{*i}^* requires computing the inversion $H^{(i)-1}$, which takes $O(n^3)$. Since W^* has n columns, the total computation cost is $O(n^4)!$ This complexity is impractical for real world datasets with large n .

Question: Can we design a more efficient algorithm for computing W^* when $b > 0$?

The answer is **YES**. We will show an algorithm that reduces the complexity to $O(n^3)$.

A Fast Algorithm for Computing $b > 0$ Solutions

Note that we can decompose $G^{(1)}, G^{(2)}, \dots, G^{(n)}$ as follows:

$$G^{(1)} \begin{bmatrix} \times & * & * & * & * \\ * & + & \circ & \circ & \circ \\ * & \circ & + & \circ & \circ \\ * & \circ & \circ & + & \circ \\ * & \circ & \circ & \circ & + \end{bmatrix} = \begin{bmatrix} + & \circ & \circ & \circ & \circ \\ \circ & + & \circ & \circ & \circ \\ \circ & \circ & + & \circ & \circ \\ \circ & \circ & \circ & + & \circ \\ \circ & \circ & \circ & \circ & + \end{bmatrix} + \begin{bmatrix} \Delta \\ \diamond \\ \diamond \\ \diamond \\ \diamond \end{bmatrix} + \begin{bmatrix} & & & & \\ & \diamond & \diamond & \diamond & \diamond \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}$$

$$G^{(2)} \begin{bmatrix} + & * & \circ & \circ & \circ \\ * & \times & * & * & * \\ \circ & * & + & \circ & \circ \\ \circ & * & \circ & + & \circ \\ \circ & * & \circ & \circ & + \end{bmatrix} = \begin{bmatrix} + & \circ & \circ & \circ & \circ \\ \circ & + & \circ & \circ & \circ \\ \circ & \circ & + & \circ & \circ \\ \circ & \circ & \circ & + & \circ \\ \circ & \circ & \circ & \circ & + \end{bmatrix} + \begin{bmatrix} \diamond \\ \Delta \\ \diamond \\ \diamond \\ \diamond \end{bmatrix} + \begin{bmatrix} & & & & \\ \diamond & & & & \\ & \diamond & \diamond & \diamond & \\ & & & & \\ & & & & \end{bmatrix}$$

.....

$$G^{(n)} \begin{bmatrix} + & \circ & \circ & \circ & * \\ \circ & + & \circ & \circ & * \\ \circ & \circ & + & \circ & * \\ \circ & \circ & \circ & + & * \\ * & * & * & * & \times \end{bmatrix} = \begin{bmatrix} + & \circ & \circ & \circ & \circ \\ \circ & + & \circ & \circ & \circ \\ \circ & \circ & + & \circ & \circ \\ \circ & \circ & \circ & + & \circ \\ \circ & \circ & \circ & \circ & + \end{bmatrix} + \begin{bmatrix} \diamond \\ \diamond \\ \diamond \\ \diamond \\ \Delta \end{bmatrix} + \begin{bmatrix} & & & & \\ & \diamond & \diamond & \diamond & \\ & & & & \\ & & & & \\ \diamond & \diamond & \diamond & \diamond & \end{bmatrix}$$

$$\begin{aligned} \times &: (1-p)b^2, & * &: (1-p)^2b^2, & + &: (1-p)pa^2 + (1-p)^2b^2, & \circ &: (1-p)^2pa^2 + (1-p)^3b^2 \\ \Delta &: -(1-p)p(a^2 - b^2), & \diamond &: -(1-p)^2p(a^2 - b^2) \end{aligned}$$

A Fast Algorithm for Computing $b > 0$ Solutions

$$\begin{array}{l}
 G^{(1)} \begin{bmatrix} \times & * & * & * & * \\ * & + & \circ & \circ & \circ \\ * & \circ & + & \circ & \circ \\ * & \circ & \circ & + & \circ \\ * & \circ & \circ & \circ & + \end{bmatrix} = \boxed{\begin{bmatrix} + & \circ & \circ & \circ & \circ \\ \circ & + & \circ & \circ & \circ \\ \circ & \circ & + & \circ & \circ \\ \circ & \circ & \circ & + & \circ \\ \circ & \circ & \circ & \circ & + \end{bmatrix}} + \boxed{\begin{bmatrix} \Delta \\ \diamond \\ \diamond \\ \diamond \\ \diamond \end{bmatrix}} + \boxed{\begin{bmatrix} & & & & \\ & \diamond & \diamond & \diamond & \diamond \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}} \\
 \\
 G^{(2)} \begin{bmatrix} + & * & \circ & \circ & \circ \\ * & \times & * & * & * \\ \circ & * & + & \circ & \circ \\ \circ & * & \circ & + & \circ \\ \circ & * & \circ & \circ & + \end{bmatrix} = \boxed{\begin{bmatrix} + & \circ & \circ & \circ & \circ \\ \circ & + & \circ & \circ & \circ \\ \circ & \circ & + & \circ & \circ \\ \circ & \circ & \circ & + & \circ \\ \circ & \circ & \circ & \circ & + \end{bmatrix}} + \boxed{\begin{bmatrix} & \diamond \\ & \Delta \\ & \diamond \\ & \diamond \\ & \diamond \end{bmatrix}} + \boxed{\begin{bmatrix} & & & & \\ \diamond & & & & \\ & \diamond & \diamond & \diamond & \\ & & & & \end{bmatrix}} \\
 \dots \\
 G^{(n)} \begin{bmatrix} + & \circ & \circ & \circ & * \\ \circ & + & \circ & \circ & * \\ \circ & \circ & + & \circ & * \\ \circ & \circ & \circ & + & * \\ * & * & * & * & \times \end{bmatrix} = \boxed{\begin{bmatrix} + & \circ & \circ & \circ & \circ \\ \circ & + & \circ & \circ & \circ \\ \circ & \circ & + & \circ & \circ \\ \circ & \circ & \circ & + & \circ \\ \circ & \circ & \circ & \circ & + \end{bmatrix}} + \boxed{\begin{bmatrix} & & & & \diamond \\ & & & & \diamond \\ & & & & \diamond \\ & & & & \diamond \\ & & & & \Delta \end{bmatrix}} + \boxed{\begin{bmatrix} & & & & \\ & & & & \\ & & & & \\ \diamond & \diamond & \diamond & \diamond & \end{bmatrix}} \\
 \text{Fixed} \qquad \qquad \qquad \text{Rank-1 Perturbation} \qquad \qquad \text{Rank-1 Perturbation}
 \end{array}$$

This special decomposition structure inspires us to use a more specialized matrix inversion algorithm to reduce the computational complexity.

A Fast Algorithm for Computing $b > 0$ Solutions

Miller's Matrix Inverse Theorem [2]: Let G and $G + Q$ be non-singular matrices. Suppose Q is of rank r and can be decomposed as $Q = E_1 + E_2 + \dots + E_r$, where each E_k is of rank 1, and $P_{k+1} = G + E_1 + E_2 + \dots + E_k$ is non-singular for $k = 1, 2, \dots, r$. Let $P_1 = G$, then

$$P_{k+1}^{-1} = P_k^{-1} - \frac{1}{1 + \text{tr}(P_k^{-1}E_k)} P_k^{-1} E_k P_k^{-1}$$

The computation of $H^{(1)^{-1}}, H^{(2)^{-1}}, \dots, H^{(n)^{-1}}$ is a special case of Miller's theorem with $r = 2$. Based on this, we design the following fast algorithm for computing $W_{*i}^* = H^{(i)^{-1}} v^{(i)}$ for all i , with a total cost $O(n^3)$.

Fast Algorithm: First, precompute these matrices

$$R^T R, H_0^{-1} = (G_0 \odot R^T R)^{-1}, [H_0^{-1} v^{(1)}, H_0^{-1} v^{(2)}, \dots, H_0^{-1} v^{(n)}] = H_0^{-1} (U \odot R^T R), \\ [H_0^{-1} e_1^{(1)}, H_0^{-1} e_1^{(2)}, \dots, H_0^{-1} e_1^{(n)}] = H_0^{-1} (G_1 \odot R^T R), [e_2^{(1)}, e_2^{(2)}, \dots, e_2^{(n)}] = G_2 \odot R^T R$$

Then for $i = 1, 2, \dots, n$, compute each $W_{*i}^* = H^{(i)^{-1}} v^{(i)}$ as follows:

$$r = H_0^{-1} v^{(i)}, \quad w = H_0^{-1} e_1^{(i)}, \quad s = r - \frac{1}{1 + w_i} r_i w, \quad t = (H_0^{-1})_{*i} - \frac{(H_0^{-1})_{ii}}{1 + w_i} w,$$

$$H^{(i)^{-1}} v^{(i)} = s - \frac{1}{1 + e_2^{(i)T} t} (e_2^{(i)T} s) t$$

Closed-Form Solutions of DEQL: $b = 0$ Case

When $b = 0$, the optimal solution of DEQL is given by

$$W_{*i,-i}^* = (H_{-i}^{(i)})^{-1} v_{-i}^{(i)} \quad \text{and} \quad W_{ii}^* \in \mathbb{R} \quad \text{for } i = 1, 2, \dots, n \quad (6)$$

- $H_{-i}^{(i)}$ is obtained by removing the i th row and column of $H^{(i)}$.
- $v_{-i}^{(i)}$ is obtained by removing the i th row of $v^{(i)}$.

This implies that, the optimal solutions for $b = 0$ have infinite many; they share the same off-diagonal and allow arbitrary diagonal.

Furthermore, we show that the EDLAE solution corresponds to one of these by choosing a zero diagonal:

Theorem 2

The EDLAE solution (2) is equivalent to the DEQL solution of the $b = 0$ case (6) with $W_{ii}^* = 0$ for all i .

Since any model in the solution space – whether with a zero diagonal or not – equally minimizes the EDLAE objective, a zero-diagonal solution does not necessarily yield better test performance than a non-zero-diagonal one.

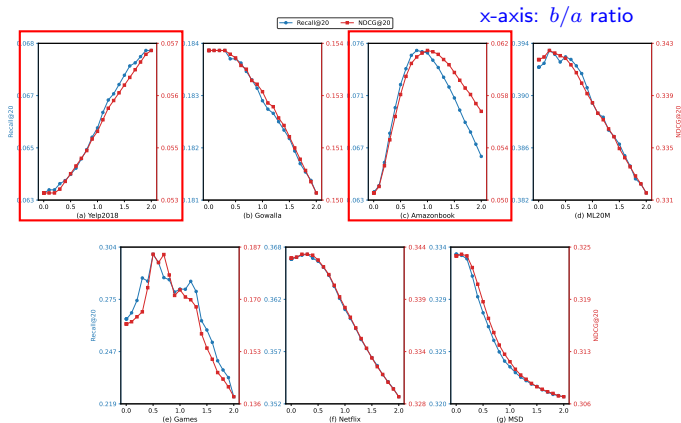
Experiment: Comparison with SOTA LAE Recommenders

| Model | Amazon-Books | | Yelp2018 | | Gowalla | |
|-----------------------------------|---------------|---------------|---------------|---------------|---------------|---------------|
| | R@20 | N@20 | R@20 | N@20 | R@20 | N@20 |
| Deep learning based models | | | | | | |
| PinSage | 0.0282 | 0.0219 | 0.0471 | 0.0393 | 0.1380 | 0.1196 |
| LightGCN | 0.0411 | 0.0315 | 0.0649 | 0.0530 | 0.1830 | 0.1554 |
| DGCF | 0.0422 | 0.0324 | 0.0654 | 0.0534 | 0.1842 | 0.1561 |
| SGL-ED | 0.0478 | 0.0379 | 0.0675 | 0.0555 | – | – |
| SimpleX | 0.0583 | 0.0468 | 0.0701 | 0.0575 | 0.1872 | 0.1557 |
| SSM (MF) | 0.0473 | 0.0367 | 0.0509 | 0.0404 | 0.1231 | 0.0878 |
| SSM (GNN) | 0.0590 | 0.0459 | 0.0737 | 0.0609 | 0.1869 | 0.1571 |
| LAE-based models | | | | | | |
| DIAE | 0.0751 | 0.0610 | 0.0678 | 0.0570 | 0.1839 | 0.1533 |
| EASE | 0.0710 | 0.0566 | 0.0657 | 0.0552 | 0.1765 | 0.1467 |
| EDLAE | 0.0711 | 0.0566 | 0.0673 | 0.0565 | 0.1844 | 0.1539 |
| ELSA | 0.0719 | 0.0594 | 0.0629 | 0.0541 | 0.1755 | 0.1490 |
| DEQL(plain) | 0.0695 | 0.0537 | 0.0647 | 0.0543 | 0.1749 | 0.1453 |
| DEQL(L2+zero-diag) | 0.0711 | 0.0567 | 0.0672 | 0.0565 | 0.1844 | 0.1539 |
| DEQL(L2) | 0.0751 | 0.0613 | 0.0685 | 0.0576 | 0.1845 | 0.1540 |

DEQL(L2): DEQL with L_2 regularization, DEQL(L2+zero-diag): DEQL with L_2 regularization and zero diagonal constraint.

- The original EDLAE solution ($b = 0$) is not optimal. DEQL solutions with $b > 0$ achieves higher test performance.
- L_2 regularization is essential for improving performance.
- The zero-diagonal constraint does not necessarily lead to better performance.

Experiment: Sensitivity Analysis of b



- The overall results show that models with $b > 0$ can achieve better performance than those with $b = 0$, highlighting the effectiveness of DEQL.
- The results on datasets **Amazonbook** and **Yelp2018** are **counter-intuitive**: the peak model performance occurs at $b > a$ (emphasize the remained entries rather than dropped ones), which is contrary to the EDLAE assumption $a \geq b$. One possible reason is that on these datasets, item interactions are extremely sparse; choosing $a > b$ forces the model to learn unreliable cross-item correlations, which degrades performance.

Conclusion

DEQL facilitates closed-form solution analysis and reveals new theoretical insights:

- When $b = 0$, the closed-form minimizer is not unique – solutions share identical off-diagonal entries while allowing arbitrary diagonal.
- When $b > 0$, a unique closed-form solutions always exists, including the previously unexplored region $b > a$.

Experiments show that

- Solutions with $b > 0$ consistently outperform the original EDLAE solutions with $b = 0$, as well as other recent LAE-based and deep learning-based recommender models.
- The previously assumed constraint $a \geq b$ does not universally guarantee optimal performance.

One future direction is to use DEQL (4) to design alternative objectives by choosing distributions $\{\mathcal{D}^{(i)}\}_{i=1}^n$ that more closely align with the MSE testing criterion than EDLAE (3), which may enhance recommendation performance.

References

- [1] Harald Steck. Autoencoders that don't overfit towards the identity. NeurIPS, 2020.
- [2] Kenneth S Miller. On the inverse of the sum of matrices. Mathematics magazine, 54(2):67–72, 1981.

Check **our paper** for more details!

<https://openreview.net/pdf?id=ANH044Wdje>

Thank you for attention!