GENERALIZING LINEAR AUTOENCODER RECOMMENDERS WITH DECOUPLED EXPECTED QUADRATIC LOSS

Anonymous authors
Paper under double-blind review

ABSTRACT

Linear autoencoders (LAEs) have gained increasing popularity in recommender systems due to their simplicity and strong empirical performance. Most LAE models, including the Emphasized Denoising Linear Autoencoder (EDLAE) introduced by (Steck, 2020), use quadratic loss during training. However, the original EDLAE only provides closed-form solutions for the hyperparameter choice $b = 0$, which limits its capacity. In this work, we generalize EDLAE objective function into a Decoupled Expected Quadratic Loss (DEQL). We show that DEQL simplifies the process of deriving EDLAE solutions and reveals solutions in a broader hyperparameter range $b > 0$, which were not derived in Steck's original paper. Additionally, we propose an efficient algorithm based on Miller's matrix inverse theorem to ensure the computational tractability for the $b > 0$ case. Empirical results on benchmark datasets show that the $b > 0$ solutions provided by DEQL outperform the $b = 0$ EDLAE baseline, demonstrating that DEQL expands the solution space and enables the discovery of models with better testing performance.

1 INTRODUCTION

In recent years, deep learning has emerged as the dominant paradigm in recommendation systems, leading to increasingly complex models. However, a growing body of empirical evidence reveals a surprising trend: simple linear models often perform comparably to, or even outperform, their deep learning counterparts (Dacrema et al., 2019). In particular, linear autoencoder-based methods such as SLIM (Ning & Karypis, 2011), EASE (Steck, 2019), EDLAE (Steck, 2020), ELSA(Vančura et al., 2022), RALE and RDLAE (Moon et al., 2023) have demonstrated strong performance, often exceeding that of more sophisticated deep models (Dacrema et al., 2021).

These methods typically aim to learn an item-to-item similarity matrix $W \in \mathbb{R}^{n\times n}$ to reconstruct the binary user-item interaction matrix $R \in \{0,1\}^{m\times n}$ with m users and n items. Each row R_{i*} encodes the interactions of user i; $R_{ij} = 1$ indicates that user i has interacted with item j, while $R_{ij} = 0$ indicates no interaction. The reconstruction takes the form RW, or equivalently $R_{i*}W$ for each user, and can be interpreted as a linear autoencoder (LAE), where W serves as both encoder and decoder. One representative example is EASE (Steck, 2019), in which W is obtained by minimizing the following objective function

$$f(W) = \|R - RW\|_F^2 \quad \text{s.t. } \text{diag}(W) = 0 \tag{1}$$

The zero diagonal constraint $\text{diag}(W) = 0$ is imposed to prevent W from overfitting towards identity. Moreover, since the prediction of each interaction R_{ij} is a weighted sum $R_{i*}W_{*j} = \sum_{k=1}^{n} R_{ik}W_{kj}$, the zero diagonal constraint enforces $W_{jj} = 0$, such that the prediction becomes $R_{i*}W_{*j} = \sum_{k=1,k\neq j}^{n} R_{ik}W_{kj}$. This means that the target R_{ij} is masked out during prediction, preventing the model from trivially using R_{ij} to predict itself. This constraint distinguishes LAEs from standard linear regression models and is widely adopted in models like EDLAE (Steck, 2020) and ELSA (Vančura et al., 2022).

Despite their empirical success, these models optimize squared error on observed entries during training, without explicitly considering the statistical nature of the evaluation process: In training,

observed entries are treated as fixed values to be reconstructed; in evaluation, especially in the strong/weak generalization settings (Steck, 2019; Moon et al., 2023), the model is evaluated on randomly masked interactions in the test set. This motivates adopting a statistical viewpoint to redesign the training objective, where interactions are treated as random variables sampled from a distribution and the objective is defined in expectation, thereby aligning with the testing scenario.

EDLAE (Steck, 2020) provides an important precursor in this direction. By introducing *dropout* and an *emphasis weighting* scheme, EDLAE effectively reshapes the loss to penalize reconstruction of masked entries more heavily, thereby reducing overfitting to the identity function, see Eq (3). We observe that considering the dropout matrix Δ as a random variable allows the objective to be expressed as an expected quadratic loss

$$f(W) = \mathbb{E}_{\Delta}\left[\|A \odot (R - (\Delta \odot R)W)\|_F^2\right], \tag{2}$$

$A \in \{a, b\}^{m\times n}$ is an emphasis matrix where a, b are hyperparameters. $\Delta \odot R$ represents random dropout applied to the fixed interaction matrix, mirroring the evaluation procedure. More importantly, $\Delta \odot R$ can itself be viewed as a random interaction matrix, providing the key insight that the objective can be *generalized* to random interaction matrices following arbitrary distributions.

Although it improves empirical performance, the theoretical framework underlying this objective's construction remains largely underexplored. In particular, while Eq (3) is claimed to be applicable for $b \geq 0$, Steck (2020) only provides the solution for $b = 0$ case, which limits its capacity. Motivated by this gap, this paper investigates how to obtain a closed-form solution for the EDLAE objective under the full hyperparameter range $b \geq 0$, and how to compute these solutions efficiently.

First, we generalize the EDLAE objective Eq (3) into a **Decoupled Expected Quadratic Loss (DEQL)** and derive its closed-form minimizer, which subsumes EDLAE as a special case. This generalization not only simplifies the derivation of EDLAE solutions but also extends them to the previously unexplored regime of $b > 0$ (Steck, 2020) (Section 3).

Next, we find that the direct solutions for $b > 0$ have an $O(n^4)$ computational complexity, which is prohibitively expensive for large-scale recommendation tasks. To overcome this challenge, we develop an efficient algorithm based on Miller's matrix inverse theorem (Miller, 1981), reducing the complexity to $O(n^3)$. This makes computing solutions for the $b > 0$ case practical (Section 4).

Finally, we evaluate solutions derived from DEQL on real-world benchmark datasets. Our experiments demonstrate that solutions with $b > 0$ consistently outperform the original EDLAE solutions with $b = 0$, confirming that expanding the solution space leads to models with stronger testing performance (Section 5).

We emphasize that DEQL is a general statistical loss function, of which EDLAE is only one special case. Beyond providing theoretical clarity, DEQL offers a unifying framework for designing new objectives for linear autoencoder (LAE) models, potentially enabling further advances in recommendation system design.

The proofs of all theorems, lemmas and propositions are presented in Appendix A. Related works are in Appendix C. Discussions are in Appendix E.

2 PRELIMINARIES

Implicit and Explicit Recommenders: Recommendation algorithms can generally be divided into two categories: explicit and implicit methods. Explicit approaches focus on predicting unseen numerical ratings that users might assign to items, whereas implicit approaches aim to predict user behaviors such as clicks, add-to-cart actions, or purchases (Steck, 2019; Dacrema et al., 2019). In this study, we focus on the implicit recommendation setting due to its greater economic significance. Let n denote the number of items and m the number of users. We are given a binary interaction matrix $R \in \{0, 1\}^{m\times n}$, where each entry $R_{i,j} = 1$ if user i has interacted with item j (e.g., through a purchase or rating), and 0 otherwise. We note that despite the difference between explicit and implicit settings; for recommendation models, both objectives aim to recover a real-valued score matrix $\hat{R} \in \mathbb{R}^{m\times n}$. The performance of the model for implicit setting is typically evaluated using information retrieval metrics such as Top-k Recall/Accuracy or Normalized Discounted Cumulative Gain (nDCG) on test set.

EDLAE Recommender System (Steck, 2020): Let $\Delta \in \{0,1\}^{m\times n}$ be a random matrix where each Δ_{ij} is i.i.d. drawn from the Bernoulli distribution such that $P(\Delta_{ij}=0)=p$ and $P(\Delta_{ij}=1)=1-p$. Let $\Delta^{(k)}$ denote a realization of Δ, and let $\odot$ denote the Hadamard (element-wise) product, so that $\Delta^{(k)}\odot R$ applies dropout element-wise to R. Define the emphasis matrix $A^{(k)}$ where $A^{(k)}_{ij}=a$ if $\Delta^{(k)}_{ij}=0$ and $A^{(k)}_{ij}=b$ if $\Delta^{(k)}_{ij}=1$. Then, the EDLAE model is obtained by optimizing the following objective function:

$$W^* = \text{argmin}_W \lim_{N\to\infty} \frac{1}{N}\sum_{k=1}^{N} \|A^{(k)}\odot(R-(\Delta^{(k)}\odot R)W)\|_F^2 \tag{3}$$

under the hyperparameters a, b, p. Since the squared Frobenius norm in Eq (3) can be expanded into the sum of weighted quadratic loss $\sum_{i=1}^{m}\sum_{j=1}^{n} {A^{(k)}_{ij}}^2 (R_{ij}-(\Delta^{(k)}_{i*}\odot R_{i*})W_{*j})^2$, if R_{ij} is dropped, its reconstruction loss $(R_{ij}-(\Delta^{(k)}_{i*}\odot R_{i*})W_{*j})^2$ is weighted by a^2; otherwise, it is weighted by b^2. The hyperparameters a, b are typically set to $a\geq b\geq 0$, thereby placing greater emphasis on dropped entries to prioritize reducing loss.

The original EDLAE paper (Steck, 2020) provides a closed-form solution to Eq (3) for the case $b=0$ and under the zero-diagonal constraint $\text{diag}(W^*)=0$, expressed as

$$W^* = \frac{1}{1-p}\left(I - C\cdot(I\odot C)^{-1}\right), \text{ where } C=\left(R^TR+\frac{p}{1-p}I\odot R^TR\right)^{-1} \tag{4}$$

While Eq (3) remains valid and meaningful for $b>0$, the solution for this case is not addressed in the original work.

3 Decoupled Expected Quadratic Loss for Linear Autoencoders

In the EDLAE optimization problem Eq (3), let $\mathcal{B}$ denote the multivariate Bernoulli distribution of Δ, then by the law of large numbers, Eq (3) can be rewritten as

$$W^* = \text{argmin}_W\, l_{\mathcal{B}}(W), \quad \text{where}$$

$$l_{\mathcal{B}}(W) = \mathbb{E}_{\Delta\sim\mathcal{B}}\left[\|A\odot(R-(\Delta\odot R)W)\|_F^2\right] = \sum_{i=1}^{n}\mathbb{E}_{\Delta\sim\mathcal{B}}\left[\|A_{*i}\odot(R_{*i}-(\Delta\odot R)W_{*i})\|_F^2\right]$$

$$= \sum_{i=1}^{n}\mathbb{E}_{\Delta\sim\mathcal{B}}\left[\|A^{(i)}R_{*i}-A^{(i)}(\Delta\odot R)W_{*i})\|_F^2\right] \tag{5}$$

Here we denote $A^{(i)}=\text{diagMat}(A_{*i})$. Since R is constant while both Δ and $A^{(i)}$ are random, define $Y^{(i)}=A^{(i)}R$ and $X^{(i)}=A^{(i)}(\Delta\odot R)$, then both $X^{(i)}$ and $Y^{(i)}$ are random. If we in further denote $\mathcal{D}^{(i)}$ as the distribution of the pair $(X^{(i)},Y^{(i)})$, then the objective function in Eq (5) can be written as

$$l_{\mathcal{B}}(W) = \sum_{i=1}^{n}\mathbb{E}_{(X^{(i)},Y^{(i)})\sim\mathcal{D}^{(i)}}\left[\|Y^{(i)}-X^{(i)}W_{*i}\|_F^2\right] \tag{6}$$

Eq (6) *decouples* W into columns, with each column W_{*i} appearing in an *expected quadratic loss* $\mathbb{E}_{(X^{(i)},Y^{(i)})\sim\mathcal{D}^{(i)}}\left[\|Y^{(i)}-X^{(i)}W_{*i}\|_F^2\right]$. This formulation is general since each $\mathcal{D}^{(i)}$ can be any distribution, while Eq (5) is a special case where $X^{(i)}$ and $Y^{(i)}$ follow distributions induced by applying random dropout to constants. We first derive the general closed-form solution of optimizing Eq (6), then specialize it to EDLAE, and show that this reformulation simplifies the analysis and reveals a broader class of solutions for $b\geq 0$ compared Steck's original solution for $b=0$ (Steck, 2020).

3.1 Decoupled Expected Quadratic Loss and its Closed-form Solution

We formally define Eq (6) as follows:

Definition 3.1. Given a set of joint distributions $\boldsymbol{\mathcal{D}} = \{\mathcal{D}^{(i)}\}_{i=1}^{n}$ over the pair (X, Y), the **decoupled expected quadratic loss** is defined as

$$
\begin{aligned}
l_{\boldsymbol{\mathcal{D}}}(W) &= \sum_{i=1}^{n} h^i_{\mathcal{D}^{(i)}}(W_{*i}), \text{ where} \\
h^i_{\mathcal{D}^{(i)}}(W_{*i}) &= \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}\left[\|Y_{*i} - XW_{*i}\|_F^2\right] \\
&= W_{*i}^T \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}\left[X^T X\right] W_{*i} - 2W_{*i}^T \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}\left[X^T Y_{*i}\right] + \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}\left[Y_{*i}^T Y_{*i}\right]
\end{aligned} \tag{7}
$$

Note that each $h^i_{\mathcal{D}^{(i)}}$ is a quadratic function of W_{*i}. Since $\mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}\left[X^T X\right]$ is positive semi-definite (as $X^T X$ is a random variable whose realizations are always positive semi-definite matrices), $h^i_{\mathcal{D}^{(i)}}$ is convex for any i. Hence, let $W^* = \text{argmin}_W\, l_{\boldsymbol{\mathcal{D}}}(W)$, then $W^*_{*i} = \text{argmin}_{W_{*i}}\, h^i_{\mathcal{D}^{(i)}}(W_{*i})$ for all i. Furthermore, if $\mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}\left[X^T X\right]$ is positive definite for all i, so that its inverse exists, then W^* can be computed as

$$
W^*_{*i} = \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}\left[X^T X\right]^{-1} \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}\left[X^T Y_{*i}\right] \text{ for } i = 1, 2, ..., n \tag{8}
$$

Eq (7) represents a general quadratic loss. A special case arises when taking $\mathcal{D} := \mathcal{D}^{(1)} = \mathcal{D}^{(2)} = ... = \mathcal{D}^{(n)}$, in which case Eq (7) reduces to

$$
l_{\boldsymbol{\mathcal{D}}}(W) = \mathbb{E}_{(X,Y)\sim\mathcal{D}}\left[\|Y - XW\|_F^2\right]
$$

Moreover, under certain condition that $\mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}\left[X^T X\right]$ is independent of i for all i, Eq (7) with low-rank constraints on W has a closed-form solution, which is discussed in Appendix B.

3.2 Adaptation to EDLAE

This section derives the closed-from solution for EDLAE from Eq (7), which covers the case $b \geq 0$. We show that our solution is equivalent to Steck's solution (Steck, 2020) for $b = 0$, and it extends to the $b > 0$ case, which was not addressed in Steck's work.

Remember that Eq (5) is a special case of Eq (7) by taking $X = A^{(i)}(\Delta \odot R)$ and $Y_{*i} = A^{(i)} R_{*i}$. By Eq (8), the solution of Eq (5) is given by

$$
W^*_{*i} = {H^{(i)}}^{-1} v^{(i)} \text{ for } i = 1, 2, ..., n, \text{ where} \tag{9}
$$

$$
H^{(i)} = \mathbb{E}_{\Delta\sim\mathcal{B}}\left[(\Delta \odot R)^T {A^{(i)}}^2 (\Delta \odot R)\right],\ v^{(i)} = \mathbb{E}_{\Delta\sim\mathcal{B}}\left[(\Delta \odot R)^T {A^{(i)}}^2 R_{*i}\right] \tag{10}
$$

The following lemma enables explicit computation of the expectations in Eq (10):

Lemma 3.2. *The $H^{(i)}$ and $v^{(i)}$ in Eq (10) can be expressed as $H^{(i)} = G^{(i)} \odot R^T R$ and $v^{(i)} = u^{(i)} \odot R^T R_{*i}$, where $G^{(i)} \in \mathbb{R}^{n\times n}$ and $u^{(i)} \in \mathbb{R}^n$ satisfy*

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

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

Furthermore, the computation of Eq (9) requires the ${H^{(i)}}^{-1}$, which exists only if $H^{(i)}$ is positive definite. The following theorem establishes sufficient conditions to ensure this property.

Theorem 3.3. *For any a, b satisfying $a \geq b > 0$, $G^{(i)}$ is positive definite. Furthermore, $H^{(i)}$ is positive definite if $G^{(i)}$ is positive definite and no column of R is a zero vector.*

Therefore, $b > 0$ serves as a sufficient condition for the validity of Eq (9), implying that the optimal W^* can be obtained using Eq (9) when $b > 0$.

Now we discuss the case when $b = 0$. In this setting, both the i-th row and i-th column of $H^{(i)}$ are zero, and the i-th row of $v^{(i)}$ is also zero. Consequently, $H^{(i)}$ is singular and its inverse ${H^{(i)}}^{-1}$ does not exist, making Eq (8) inapplicable for computing the optimal W^*.

To proceed, we define submatrices and subvectors using the subscript $-i$ notation: if Q is an $n \times n$ matrix, then Q_{-i} is a $(n-1) \times (n-1)$ matrix obtained by removing the i-th row and i-th column of Q; if q is an n dimensional vector, then q_{-i} is an $n-1$ dimensional vector obtained by removing q_i from q. Under this notation, we can write $H^{(i)}_{-i} = G^- \odot (R^T R)_{-i}$, where G^- is an $(n-1) \times (n-1)$ matrix with diagonal elements $(1-p)pa^2$ and off-diagonal elements $(1-p)^2 pa^2$. It is easy to verify that G^- is positive definite, hence $H^{(i)}_{-i}$ is positive definite. Likewise, $v^{(i)}_{-i} = u^- \odot (R^T R_{*i})_{-i}$, where u^- is an $n-1$ dimensional vector with all elements being $(1-p)pa^2$.

Denote the vector $(W_{*i})_{-i}$ as $W_{*i,-i}$, then Eq (5) can be written as

$$l_{\mathcal{B}}(W) = \sum_{i=1}^{n} W_{*i}^T H^{(i)} W_{*i} - 2W_{*i}^T v^{(i)} + \mathbb{E}_{\Delta \sim \mathcal{B}} \left[R_{*i}^T {A^{(i)}}^2 R_{*i} \right] \tag{11}$$

$$= \sum_{i=1}^{n} W_{*i,-i}^T H^{(i)}_{-i} W_{*i,-i} - 2W_{*i,-i}^T v^{(i)}_{-i} + \mathbb{E}_{\Delta \sim \mathcal{B}} \left[R_{*i}^T {A^{(i)}}^2 R_{*i} \right] \tag{12}$$

Therefore, the solution $W^* = \text{argmin}_W \, l_{\mathcal{B}}(W)$ is expressed as

$$W^*_{*i,-i} = (H^{(i)}_{-i})^{-1} v^{(i)}_{-i} \text{ and } W^*_{ii} \in \mathbb{R} \quad \text{for } i = 1, 2, ..., n \tag{13}$$

Eq (13) reveals that, the optimal W^* is not unique, belongs to an infinite set of solutions. All such solutions share the same off-diagonal entries, while the diagonal elements can take arbitrary values. The following theorem shows that Steck's solution Eq (4) is a special case of Eq (13), corresponding to the choice of zero diagonal.

Theorem 3.4. *Suppose no column of R is a zero vector. If taking $W_{ii} = 0$ for all i in Eq (13), then Eq (13) and Eq (4) are equivalent.*

It is important to note that varying the diagonal of W^* can lead to different performance on test data, and Eq (13) does not provide theoretical guidance on which choice of diagonal elements gives the best performance. However, empirical results suggest that a W^* with non-zero diagonal elements can outperform the zero-diagonal solution in certain cases (Moon et al., 2023).

3.3 Adding L_2 Regularizer and Zero-diagonal Constraint

In LAE-based recommender systems, L_2 regularizer and zero diagonal constraint are commonly applied to the objective function, as they are established techniques for improving test performance. This section discusses the closed-form solution of the optimization problem Eq (5) when the L_2 regularizer or the zero-diagonal constraint is applied.

Adding L_2 Regularizer: Given $\lambda > 0$, Eq (5) with L_2 regularizer is expressed as

$$W^* = \text{argmin}_W \, l_{\mathcal{B}}(W) + \lambda \|W\|_F^2 \tag{14}$$

Adding Zero-diagonal Constraint: Eq (5) with zero-diagonal constraint is expressed as

$$W^* = \text{argmin}_W \, l_{\mathcal{B}}(W) \quad \text{s.t. } \text{diag}(W) = 0 \tag{15}$$

In these cases, the solution Eq (9) is modified accordingly, as presented below.

Proposition 3.5. *(a) The solution of Eq (14) is*

$$W^*_{*i} = \left(H^{(i)} + \lambda I \right)^{-1} v^{(i)} \quad \textit{for } i = 1, 2, ..., n \tag{16}$$

(b) The solution of Eq (15) is

$$W^*_{*i} = {H^{(i)}}^{-1} v^{(i)} - \frac{({H^{(i)}}^{-1} v^{(i)})_i}{({H^{(i)}}^{-1} l^{(i)})_i} {H^{(i)}}^{-1} l^{(i)} \quad \textit{for } i = 1, 2, ..., n \tag{17}$$

where $l^{(i)}$ is an n-dimensional vector with $l^{(i)}_i = 1$ and $l^{(i)}_j = 0$ for all $j \neq i$.

4 AN EFFICIENT ALGORITHM FOR THE CLOSED-FORM SOLUTION

Recall from Theorem 3.3 that the optimal W^* for the $b > 0$ case of EDLAE can be computed by Eq (9). However, a major challenge with Eq (9) is its high computational complexity: since $H^{(i)}$ differs for each i, computing each inverse ${H^{(i)}}^{-1}$ costs $O(n^3)$, resulting in a total cost of $O(n^4)$ for all i, which is computationally impractical.

In this section, we propose a practical algorithm to reduce the overall complexity of computing Eq (9). Our algorithm is based on Miller's matrix inverse theorem:

Theorem 4.1. *((Miller, 1981)) Let G and $G + Q$ be non-singular matrices. Suppose Q is of rank r and can be decomposed as $Q = E_1 + E_2 + ... + E_r$, where each E_k is of rank 1, and $P_{k+1} = G + E_1 + E_2 + ... + E_k$ is non-singular for $k = 1, 2, ..., r$. Let $P_1 = G$, then*

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

In Lemma 3.2, we define $H^{(i)} = G^{(i)} \odot R^T R$, which can be decomposed as

$$H^{(i)} = G_0 \odot R^T R + G_1^{(i)} \odot R^T R + G_2^{(i)} \odot R^T R$$

where G_0 is a matrix with diagonal elements equal to $(1-p)pa^2 + (1-p)^2 b^2$ and off-diagonal elements equal to $(1-p)^2 pa^2 + (1-p)^3 b^2$; $G_1^{(i)}$ is a matrix with $(G_1^{(i)})_{ji} = -(1-p)^2 p(a^2 - b^2)$ for $j \neq i$, $(G_1^{(i)})_{ii} = -(1-p)p(a^2 - b^2)$, and all other elements zero; $G_2^{(i)}$ is a matrix with $(G_2^{(i)})_{ij} = -(1-p)^2 p(a^2 - b^2)$ for $j \neq i$ and all other elements zero.

Denote $H_0 = G_0 \odot R^T R$, $E_1^{(i)} = G_1^{(i)} \odot R^T R$ and $E_2^{(i)} = G_2^{(i)} \odot R^T R$, then $H^{(i)} = H_0 + E_1^{(i)} + E_2^{(i)}$. Note that H_0 is positive definite and independent of i, $E_1^{(i)}$ is of rank 1 with only the i-th column being nonzero, and $E_2^{(i)}$ is of rank 1 with the i-th row (excluding $(E_2^{(i)})_{ii}$) being nonzero.

Applying Theorem 4.1, ${H^{(i)}}^{-1}$ can be computed with the following two steps:

$${H_+^{(i)}}^{-1} = (H_0 + E_1^{(i)})^{-1} = H_0^{-1} - \frac{1}{1 + \text{tr}(H_0^{-1} E_1^{(i)})} H_0^{-1} E_1^{(i)} H_0^{-1} \tag{18}$$

$${H^{(i)}}^{-1} = (H_0 + E_1^{(i)} + E_2^{(i)})^{-1} = H_+^{-1} - \frac{1}{1 + \text{tr}(H_+^{-1} E_2^{(i)})} H_+^{-1} E_2^{(i)} H_+^{-1} \tag{19}$$

Let $e_1^{(i)}$ be the i-th column of $E_1^{(i)}$, ${e_2^{(i)}}^T$ be the i-th row of $E_2^{(i)}$, then Eq (18) and Eq (19) can be simplified as

$${H_+^{(i)}}^{-1} = H_0^{-1} - \frac{1}{1 + (H_0^{-1})_{i*} e_1^{(i)}} (H_0^{-1} e_1^{(i)})(H_0^{-1})_{i*} \tag{20}$$

$${H^{(i)}}^{-1} = {H_+^{(i)}}^{-1} - \frac{1}{1 + {e_2^{(i)}}^T ({H_+^{(i)}}^{-1})_{*i}} ({H_+^{(i)}}^{-1})_{*i} ({e_2^{(i)}}^T {H_+^{(i)}}^{-1}) \tag{21}$$

Observe that, given H_0^{-1}, the computation of each ${H^{(i)}}^{-1}$ using Eq (20) and Eq (21) requires only $O(n^2)$ operations, resulting in a total cost of $O(n^3)$ for all i. This significantly reduces the original $O(n^4)$ complexity of computing Eq (9).

Moreover, the computation can be further simplified by directly computing ${H^{(i)}}^{-1} v^{(i)}$ without explicitly forming ${H^{(i)}}^{-1}$. By Eq (20) and Eq (21),

$${H_+^{(i)}}^{-1} v^{(i)} = H_0^{-1} v^{(i)} - \frac{1}{1 + (H_0^{-1})_{i*} e_1^{(i)}} (H_0^{-1} e_1^{(i)}) \left[(H_0^{-1})_{i*} v^{(i)} \right] \tag{22}$$

$${H^{(i)}}^{-1} v^{(i)} = {H_+^{(i)}}^{-1} v^{(i)} - \frac{1}{1 + {e_2^{(i)}}^T ({H_+^{(i)}}^{-1})_{*i}} ({H_+^{(i)}}^{-1})_{*i} \left[({e_2^{(i)}}^T {H_+^{(i)}}^{-1}) v^{(i)} \right] \tag{23}$$

in which the scalars $(H_0^{-1})_{i*}v^{(i)}$ and $(e_2^{(i)^T}H_+^{(i)^{-1}})v^{(i)}$ can be computed first. In Eq (23), the $(H_+^{(i)^{-1}})_{*i}$ term can be computed by Eq (20),

$$(H_+^{(i)^{-1}})_{*i} = (H_0^{-1})_{*i} - \frac{(H_0^{-1})_{ii}}{1 + (H_0^{-1})_{i*}e_1^{(i)}}(H_0^{-1}e_1^{(i)})$$

Denote $s = H_+^{(i)^{-1}}v^{(i)}$, $t = (H_+^{(i)^{-1}})_{*i}$. Let U be an $n \times n$ matrix with diagonal elements equal to $(1-p)b^2$ and off-diagonal elements equal to $(1-p)pa^2 + (1-p)^2b^2$, let G_1 be an $n \times n$ matrix with diagonal elements $-(1-p)p(a^2-b^2)$ and off-diagonal elements $-(1-p)^2p(a^2-b^2)$, and let G_2 be an $n \times n$ matrix with zeros on the diagonal and off-diagonal elements $-(1-p)^2p(a^2-b^2)$. Then we can summarize our computation as follows.

Fast Algorithm for Computing Eq (9): First, precompute these matrices

$$R^TR, \;\; H_0^{-1} = \left(G_0 \odot R^TR\right)^{-1}, \;\; [v^{(1)}, v^{(2)}, ...v^{(n)}] = U \odot R^TR,$$
$$[e_1^{(1)}, e_1^{(2)}, ..., e_1^{(n)}] = G_1 \odot R^TR, \;\; [e_2^{(1)}, e_2^{(2)}, ..., e_2^{(n)}] = G_2 \odot R^TR,$$

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

$$r = H_0^{-1}v^{(i)}, \quad w = H_0^{-1}e_1^{(i)}$$
$$s = r - \frac{1}{1+w_i}r_iw, \quad t = (H_0^{-1})_{*i} - \frac{(H_0^{-1})_{ii}}{1+w_i}w$$
$$H^{(i)^{-1}}v^{(i)} = s - \frac{1}{1+e_2^{(i)^T}t}(e_2^{(i)^T}s)t$$

We now analyze the computational complexity of the above algorithm. In the precomputing stage, R^TR costs $O(mn^2)$, H_0^{-1} costs $O(n^3)$, and $U \odot R^TR, G_1 \odot R^TR, G_2 \odot R^TR$ cost $O(n^2)$ respectively. In the computing stage, each W_{*i}^* is obtained via matrix-vector or vector-vector multiplications with a complexity of $O(n^2)$, resulting in an overall complexity of $O(n^3)$ for the entire W^*. Therefore, the total complexity of the algorithm is $O\left(\max(m+n)n^2\right)$. This complexity is the same as the closed-form solutions of EASE (Steck, 2019) and EDLAE (Steck, 2020).

Complexity for Sparse Matrix: One computational bottleneck in the above algorithm is the computation of R^TR. If R is sparse and all its entries are integers, then the algorithm's complexity can be further reduced. Suppose R contains k non-zeros. The multiplication R^TR takes two matrices R^T and R as input, which together contain $2k$ non-zeros; the output is an $n \times n$ matrix containing at most n^2 nonzeros. By (Abboud et al., 2024), the complexity of computing R^TR is $O((2k+n^2)^{1.346})$. Hence, the total complexity reduces to $O((2k+n^2)^{1.346} + n^3)$, where the n^3 term comes from inverting H_0.

Adapting the Algorithm to the L2 regularizer and Zero-diagonal Constraint Cases: In Eq (16), note that $H^{(i)} + \lambda I = (H_0 + \lambda I) + E_1^{(i)} + E_2^{(i)}$, where $H_0 + \lambda I$ is independent of i. This means that we can compute Eq (16) using the above algorithm by replacing H_0 with $H_0 + \lambda I$. In Eq (17), we first compute $H^{(i)^{-1}}v^{(i)}$ with the algorithm; then, by replace $v^{(i)}$ with $l^{(i)}$, we compute $H^{(i)^{-1}}l^{(i)}$ with the same method. Once $H^{(i)^{-1}}v^{(i)}$ and $H^{(i)^{-1}}l^{(i)}$ are obtained, Eq (17) can be computed accordingly.

5 EXPERIEMENTS

This section provides experimental results comparing DEQL with state-of-the-art collaborative filtering models, including linear models and deep learning based models. Additional experiments on tim and space costs can be found in Appendix D.

5.1 EXPERIMENTAL SET-UP

Datasets: We conduct extensive experiments to verify our theorical claims. Specifically, we employ six publicly available datasets, from small to large, including Games, Beauty, Gowalla, ML-20M,

Table 1: Performance comparison and dataset statistics across six different datasets under strong generalization setting. We highlight the best results in bold. DEQL refer to Eq (9), DEQL(L2+zero-diag) refers to combining Eq (16) and Eq (17), and DEQL(L2) refers to Eq (16) solely.

Model	Games		Beauty		Gowalla		ML20M		Netflix		MSD	
	R@20	N@20	R@20	N@20	R@20	N@20	R@20	N@20	R@20	N@20	R@20	N@20
DLAE	0.2771	0.1664	0.1329	0.0886	0.2143	0.1916	0.3924	0.3409	0.3620	0.3395	0.3290	0.3210
EASE	0.2733	0.1640	0.1323	0.0875	0.2230	0.1988	0.3905	0.3390	0.3618	0.3388	0.3332	0.3261
EDLAE	0.2851	0.1681	0.1324	0.0850	0.2268	0.2012	0.3925	0.3421	0.3656	0.3427	0.3336	0.3258
DEQL	0.2524	0.1565	0.1093	0.0670	0.2149	0.1909	0.3844	0.3347	0.3606	0.3382	0.3329	0.3256
DEQL(L2+zero-diag)	0.2872	0.1704	0.1388	**0.0898**	0.2278	0.2027	0.3934	**0.3429**	0.3656	0.3423	**0.3344**	**0.3268**
DEQL(L2)	**0.2998**	**0.1842**	**0.1391**	0.0881	**0.2288**	**0.2033**	**0.3934**	0.3426	**0.3658**	**0.3428**	0.3340	0.3265
# items	896		4,394		13,681		20,108		17,769		41,140	
# users	1,006		17,971		29,243		136,677		463,435		571,353	
# inter.	15,276		75,472		677,956		9,990,682		56,880,037		33,633,450	

Netflix, and MSD (Steck, 2019; Ni et al., 2019; Seol et al., 2024) to compare with LAE based models under strong generalization setting where test users do not appear in training dataset. When comparing with modern deep learning based models, since most of these models rely on user and item embeddings to make prediction and require users appear in training dataset, we further evaluate our methods on 3 additional widely used datasets: Amazonbook, Yelp2018 and Gowalla He et al. (2020) under weak generalization setting. For better generalization, we preprocess the data following (Steck, 2020; He et al., 2020).

Baseline models and Evaluation Metrics: We compare it against the following state-of-the-art linear autoencoder-based recommendation models: EASE (Steck, 2019), DLAE (Steck, 2020), EDLAE (Steck, 2020) and recent deep learning based models: PinSage (Ying et al., 2018), LightGCN (He et al., 2020), DGCF (Wang et al., 2020), SimpleX (Mao et al., 2021), SGL-ED (Wu et al., 2021) and SSM (Wu et al., 2024a). We evaluate model performance using widely adopted ranking metrics: Recall@20 and NDCG@20.

5.2 Reproducibility

For hyper-parameter tuning, We performed a grid search to optimize the hyperparameters of the linear autoencoder models. Since in the objective function, all quadratic entries are equipped with either a or b and the solution to it is scalar invariant (e.g., in Eq 3, although $(a, b) = (1, 0.1)$ and $(a, b) = (10, 1)$ gives different Loss value, the obtained W^* will always be the same), only the ratio b/a affects the closed-form solution. Therefore, we fix $a = 1$ and search b over the range [0.1,0.25,...,2.0]. The L2 regularization coefficient is searched over [10.0,20.0,...50,100.0,300.0,500.0], and the dropout rate p is varied across [0.1,0.2,...,0.5,0.8]. All experiments are conducted on a Linux server equipped with 500 GB of memory, four NVIDIA 3090 GPUs, and a 96-core Intel(R) Xeon(R) Platinum 8268 CPU @ 2.90GHz. Our code is available at `https://anonymous.4open.science/r/ICLR2026_DEQL_new-441D/README.md`

5.3 Experimental Results

In this section, we present the experimental results over two evaluation metrics. Specifically, we would answer the following research questions:
RQ1 Can the generalized quadratic loss DEQL(L2)/DEQL(L2+zero-diag) improve recommendation performance over existing linear autoencoder-based models?
RQ2 Is the zero diagonal constraint necessary for optimal performance, or can models benefit from non-zero diagonals as suggested by DEQL(L2)?
RQ3: How does DEQL families perform compared with modern deep learning based models?

RQ1: DEQL(L2) substantially improves recommendation performance over state-of-the-art linear autoencoder models across a diverse set of benchmarks. We evaluate DEQL against three widely recognized baselines:DLAE, EASE, and EDLAE (Steck et al., 2024)(Jin et al., 2021)on six public datasets: Games, Beauty, Gowalla, ML-20M, Netflix, and MSD. As shown in Table 1, DEQL(L2) achieves the highest Recall and NDCG on 5 out of the 6 datasets, often with a significant margin. For instance, on the Games dataset, DEQL(L2) improves Recall@20 from 0.2851 (EDLAE) to 0.2998, and NDCG@20 from 0.1681 to 0.1842, showing that our framework provides better ranking quality even in small-scale, sparse interaction settings. On large-scale datasets like ML-20M, DEQL main-

Table 2: Performance comparison and dataset statistics for LAE-based model and advanced deep learning based model under weak generalization setting.

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						
DLAE	**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
DEQL	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
# items	91,599		38,048		40,981	
# users	52,643		31,668		29,858	
# inter.	2,984,108		1,561,406		1,027,370	

tains top performance (Recall@20 = 0.3934, NDCG@20 = 0.3426) that matches and even improves upon the best baseline (EDLAE: Recall@20 = 0.3925, NDCG@20 = 0.3421). This suggests that our framework retains its effectiveness when scaled to millions of interactions and hundreds of thousands of users due to its low complexity. These consistent improvements across datasets of varying sparsity and scale demonstrate the robustness of the DEQL formulation. These gains stem from our theoretical insight: by generalizing the EDLAE formulation to allow $b > 0$, our framework explores a richer solution space with closed-form solvability.

RQ2: When $b = 0$, the training loss does not constrain the diagonal of W, leading to non-uniqueness in the optimal solution. Instead of enforcing zero diagonals as in EASE/EDLAE/DEQL(L2+zero-diag), DEQL(L2) allows the diagonal to be arbitry real number, allowing us to explore richer searching space. Our empirical results support this design. On Gowalla, DEQL(L2)achieves Recall@20 = 0.2288 and NDCG@20 = 0.2033, outperforming DEQL(L+zero-diag) (0.2278 / 0.2027) and EDLAE (0.2268 / 0.2012). On ML20M, DEQL reaches Recall@20 = 0.3934 and NDCG@20 = 0.3426, closely matching DEQL(L2+zero-diag) (0.3934 / 0.3429) and surpassing EDLAE (0.3925 / 0.3421). These results demonstrate that adjusting b alone is sufficient to ensure strong generalization ability without a hard constraint as zero-diagonal constraints. These findings indicate that the diagonal constraint in EDLAE is NOT necessary and by introducing non-zero b, LAEs can still achieve competitive performance. An explanation of these results can be found in Appendix E.2.

RQ3: In Table 2 we report the results compared with recent deep learning based models. Our proposed DEQL(L2) and DEQL(L2+zero-diag) not only achieves better performance than most classic baselines, in some cases we can surpasses the most recent deep learning–based recommendation models. For example, when comparing with a recent strong benchmark SSM, DEQL(L2) attains comparable results on Yelp2018/Gowalla even superior to it on Amazonbook dataset with as large as 27% and 34% margin on R@20 and N@20 respectively. Overall, these results demonstrate the robustness and strong empirical performance of our generalized quadratic loss.

6 Conclusions

This paper aims to advance the EDLAE recommender system by extending its closed-form solution to a broader range of hyperparameter choices, and develop an efficient algorithm to compute these solutions. We first generalize the EDLAE objective function into the Decoupled Expected Quadratic Loss (DEQL), derive its closed-form solutions, and then apply them back to EDLAE. We show that, through DEQL, the original EDLAE solution for $b = 0$ can be extended to the wider range $b \geq 0$, enabling exploration of a larger solution space. To address the high computational complexity of solutions for $b > 0$, we develop an efficient algorithm based on Miller's matrix inverse theorem, reducing the complexity from $O(n^4)$ to $O(n^3)$. Experimental results demonstrate that most solutions for $b > 0$ outperform the $b = 0$ baseline, showing that DEQL expands the solution space and enables the discovery of models with better testing performance. Furthermore, DEQL is a general loss function that may inspire the construction of other specialized objectives for LAE models.

REFERENCES

Amir Abboud, Karl Bringmann, Nick Fischer, and Marvin Künnemann. The time complexity of fully sparse matrix multiplication. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 4670–4703. SIAM, 2024.

Emmanuel J Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE transactions on information theory*, 56(5):2053–2080, 2010.

Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pp. 7–10, 2016.

Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. Are we really making much progress? a worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pp. 101–109, 2019.

Maurizio Ferrari Dacrema, Simone Boglio, Paolo Cremonesi, and Dietmar Jannach. A troubling analysis of reproducibility and progress in recommender systems research. *ACM Trans. Inf. Syst.*, 39(2), January 2021.

Rina Foygel, Ohad Shamir, Nati Srebro, and Russ R Salakhutdinov. Learning with the weighted trace-norm under arbitrary sampling distributions. *Advances in neural information processing systems*, 24, 2011.

Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247*, 2017.

Xiangnan He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua. Neural collaborative filtering. In *WWW'17*, 2017.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 639–648, 2020.

Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *IEEE International Conference on Data Mining*, pp. 263–272. IEEE, 2008.

Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. *Recommender systems: an introduction*. Cambridge university press, 2010.

Ruoming Jin, Dong Li, Jing Gao, Zhi Liu, Li Chen, and Yang Zhou. Towards a better understanding of linear models for recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 776–785, 2021.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

Aravindh Krishnamoorthy and Deepak Menon. Matrix inversion using cholesky decomposition. In *2013 signal processing: Algorithms, architectures, arrangements, and applications (SPA)*, pp. 70–72. IEEE, 2013.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Dong Li, Ruoming Jin, and Bin Ren. Revisiting recommendation loss functions through contrastive learning (technical report). *arXiv preprint arXiv:2312.08520*, 2023.

Zhuang Liu, Yunpu Ma, Yuanxin Ouyang, and Zhang Xiong. Contrastive learning for recommender system. *arXiv preprint arXiv:2101.01317*, 2021.

David G Luenberger and Yinyu Ye. *Linear and nonlinear programming, 3rd Edition*. Springer, 2008.

Kelong Mao, Jieming Zhu, Jinpeng Wang, Quanyu Dai, Zhenhua Dong, Xi Xiao, and Xiuqiang He. Simplex: A simple and strong baseline for collaborative filtering. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pp. 1243–1252, 2021.

Kenneth S Miller. On the inverse of the sum of matrices. *Mathematics magazine*, 54(2):67–72, 1981.

Jaewan Moon, Hye-young Kim, and Jongwuk Lee. It's enough: Relaxing diagonal constraints in linear autoencoders for recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1639–1648, 2023.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 188–197, 2019.

Xia Ning and George Karypis. Slim: Sparse linear methods for top-N recommender systems. In *2011 IEEE 11th International Conference on Data Mining*, pp. 497–506. IEEE, 2011.

Benjamin Recht. A simpler approach to matrix completion. *Journal of Machine Learning Research*, 12(12), 2011.

Steffen Rendle. Factorization machines. In *2010 IEEE International Conference on Data Mining*, pp. 995–1000. IEEE, 2010.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. UAI '09, 2009.

Jinseok Seol, Minseok Gang, Sang-goo Lee, and Jaehui Park. Proxy-based item representation for attribute and context-aware recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pp. 616–625, 2024.

Ohad Shamir and Shai Shalev-Shwartz. Collaborative filtering with the trace norm: Learning, bounding, and transducing. In *Proceedings of the 24th Annual Conference on Learning Theory*, pp. 661–678. JMLR Workshop and Conference Proceedings, 2011.

Harald Steck. Embarrassingly shallow autoencoders for sparse data. In *The World Wide Web Conference*, pp. 3251–3257, 2019.

Harald Steck. Autoencoders that don't overfit towards the identity. In *Advances in Neural Information Processing Systems*, volume 33, pp. 19598–19608, 2020.

Harald Steck, Chaitanya Ekanadham, and Nathan Kallus. Is cosine-similarity of embeddings really about similarity? In *Companion Proceedings of the ACM Web Conference 2024*, pp. 887–890, 2024.

Juan Terven, Diana-Margarita Cordova-Esparza, Julio-Alejandro Romero-González, Alfonso Ramírez-Pedraza, and EA Chávez-Urbiola. A comprehensive survey of loss functions and metrics in deep learning. *Artificial Intelligence Review*, 58(7):195, 2025.

Vojtěch Vančura, Rodrigo Alves, Petr Kasalickỳ, and Pavel Kordík. Scalable linear shallow autoencoder for collaborative filtering. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pp. 604–609, 2022.

Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999.

Chenyang Wang, Yuanqing Yu, Weizhi Ma, Min Zhang, Chong Chen, Yiqun Liu, and Shaoping Ma. Towards representation alignment and uniformity in collaborative filtering. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 1816–1825, 2022.

Feng Wang and Huaping Liu. Understanding the behaviour of contrastive loss. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2495–2504, 2021.

Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*, pp. 1–7. 2017.

Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 165–174, 2019.

Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. Disentangled graph collaborative filtering. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pp. 1001–1010, 2020.

Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. Self-supervised graph learning for recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pp. 726–735, 2021.

Jiancan Wu, Xiang Wang, Xingyu Gao, Jiawei Chen, Hongcheng Fu, and Tianyu Qiu. On the effectiveness of sampled softmax loss for item recommendation. *ACM Transactions on Information Systems*, 42(4):1–26, 2024a.

Junkang Wu, Jiawei Chen, Jiancan Wu, Wentao Shi, Jizhi Zhang, and Xiang Wang. Bsl: Understanding and improving softmax loss for recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pp. 816–830. IEEE, 2024b.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 974–983, 2018.

Chang Zhou, Jianxin Ma, Jianwei Zhang, Jingren Zhou, and Hongxia Yang. Contrastive learning for debiased candidate generation in large-scale recommender systems. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 3985–3995, 2021.

A MATHEMATICAL PROOFS

Proof of Lemma 3.2: $H^{(i)}$ can be computed as follows. For any k, l,

$$H^{(i)}_{kl} = \mathbb{E}_\Delta[(\Delta \odot R)^T_{*k} {A^{(i)}}^2 (\Delta \odot R)_{*l}] = \mathbb{E}_\Delta[\sum_{s=1}^{m} \Delta_{sk} R_{sk} {A^{(i)}_{ss}}^2 \Delta_{sl} R_{sl}]$$
$$= \sum_{s=1}^{m} \mathbb{E}_\Delta[\Delta_{sk} R_{sk} {A^{(i)}_{ss}}^2 \Delta_{sl} R_{sl}] = \sum_{s=1}^{m} \mathbb{E}_\Delta[\Delta_{sk}\Delta_{sl} {A^{(i)}_{ss}}^2] R_{sk} R_{sl} \tag{24}$$

Note that $A^{(i)}_{ss} = A_{si}$, which depends on Δ_{si}. Since we assume each Δ_{ij} is an i.i.d. Bernoulli random variable, $\mathbb{E}_\Delta[\Delta_{sk}\Delta_{sl}A^2_{si}]$ is independent of s. Thus we can let a z be a specific value of s and rewrite Eq (24) as

$$H^{(i)}_{kl} = \mathbb{E}_\Delta[\Delta_{zk}\Delta_{zl}A^2_{zi}] \sum_{s=1}^{m} R_{sk}R_{sl} = \mathbb{E}_\Delta[\Delta_{zk}\Delta_{zl}A^2_{zi}] R^T_{*k}R_{*l}$$

Define $G^{(i)} \in \mathbb{R}^{n\times n}$ where $G^{(i)}_{kl} = \mathbb{E}_\Delta[\Delta_{zk}\Delta_{zl}A^2_{zi}]$, then $H^{(i)} = G^{(i)} \odot R^T R$. $G^{(i)}$ can be computed as follows: Given i, for any k, l,

$$\Delta_{zk}\Delta_{zl}A^2_{zi} = \begin{cases} a^2 & \text{if } \Delta_{zk}=1 \text{ and } \Delta_{zl}=1 \text{ and } \Delta_{zi}=0 \\ b^2 & \text{if } \Delta_{zk}=1 \text{ and } \Delta_{zl}=1 \text{ and } \Delta_{zi}=1 \\ 0 & \text{otherwise} \end{cases}$$

Since

$$P\left(\Delta_{zk}=1 \text{ and } \Delta_{zl}=1 \text{ and } \Delta_{zi}=0\right) = \begin{cases} (1-p)p & \text{if } k=l\neq i \\ (1-p)^2 p & \text{if } i\neq k\neq l\neq i \end{cases}$$

$$P\left(\Delta_{zk}=1 \text{ and } \Delta_{zl}=1 \text{ and } \Delta_{zi}=1\right) = \begin{cases} 1-p & \text{if } k=l=i \\ (1-p)^2 & \text{if } k=l\neq i \text{ or } k\neq l=i \text{ or } l\neq k=i \\ (1-p)^3 & \text{if } i\neq k\neq l\neq i \end{cases}$$

we have

$$G^{(i)}_{kl} = \mathbb{E}_\Delta[\Delta_{zk}\Delta_{zl}A^2_{zi}] = \begin{cases} (1-p)b^2 & \text{if } k=l=i \\ (1-p)^2 b^2 & \text{if } k\neq l=i \text{ or } l\neq k=i \\ (1-p)pa^2 + (1-p)^2 b^2 & \text{if } k=l\neq i \\ (1-p)^2 pa^2 + (1-p)^3 b^2 & \text{if } i\neq k\neq l\neq i \end{cases}$$

On the other hand, $v^{(i)}$ can be computed as follows. For any k,

$$v^{(i)}_k = \mathbb{E}_\Delta[(\Delta \odot R)^T_{*k} {A^{(i)}}^2 R_{*i}] = \mathbb{E}_\Delta[\sum_{s=1}^{m} \Delta_{sk} R_{sk} {A^{(i)}_{ss}}^2 R_{si}] = \sum_{s=1}^{m} \mathbb{E}_\Delta[\Delta_{sk}A^2_{si}] R_{sk}R_{si}$$
$$= \mathbb{E}_\Delta[\Delta_{zk}A^2_{zi}] R^T_{*k}R_{*i}$$

Define $u^{(i)} \in \mathbb{R}^n$ where $u^{(i)}_k = \mathbb{E}_\Delta[\Delta_{zk}A^2_{zi}]$, then we can write $v^{(i)} = u^{(i)} \odot R^T R_{*i}$. $u^{(i)}$ can be computed as follows: Given any k,

$$u^{(i)}_k = \mathbb{E}_\Delta[\Delta_{zk}A^2_{zi}] = \begin{cases} (1-p)b^2 & \text{if } k=i \\ (1-p)pa^2 + (1-p)^2 b^2 & \text{if } k\neq i \end{cases}$$

□

Proof of Theorem 3.3: Observe that $G^{(i)}$ can be decomposed as the sum of three matrices:

$$G^{(i)} = (1-p)^2 b^2 J + ((1-p)^2 p(a^2-b^2))J^{(i)} + \Lambda^{(i)}$$

where J is an $n \times n$ matrix of all 1s, $J^{(i)}$ is an $n \times n$ matrix that its elements on i-th row and i-th column are 0s while all other elements are 1s. $\Lambda^{(i)}$ is an $n \times n$ diagonal matrix with $\Lambda^{(i)}_{ii} = (1-p)pb^2$ and $\Lambda^{(i)}_{jj} = (1-p)p^2a^2 + (1-p)^2pb^2$ for all $j \neq i$.

It is easy to prove the positive semi-definiteness of J and $J^{(i)}$. Thus, when $a \geq b > 0$, $(1-p)^2b^2J$ and $((1-p)^2p(a^2-b^2))J^{(i)}$ are positive semi-definite. Moreover, $\Lambda^{(i)}$ is positive definite because all its diagonal elements are positive. Therefore, $G^{(i)}$ is positive definite.

R^TR is a positive semi-definite matrix. If no column of R is a zero vector, then all diagonal elements of R^TR are positive. By the Schur product theorem (Theorem 7.5.3 (b), (Horn & Johnson, 2012)), if $G^{(i)}$ is positive definite and the diagonal elements of R^TR are all positive, then $H^{(i)} = G^{(i)} \odot R^TR$ is positive definite.

□

Proof of Theorem 3.4: Since the W^* in Eq (13) has zero diagonal, we only need to verify the equivalence of non-diagonal elements. Let us write $(C_{*i})_{-i}$ as $C_{*i,-i}$, then $W_{*i,-i}$ by Eq (13) is expressed as

$$W_{*i,-i} = -\frac{1}{1-p}\frac{1}{C_{ii}}C_{*i,-i}$$

Thus, our goal is to prove

$$(H^{(i)}_{-i})^{-1}v^{(i)}_{-i} = -\frac{1}{1-p}\frac{1}{C_{ii}}C_{*i,-i} \text{ for any } i \in \{1,2,...,n\} \tag{25}$$

To show this, first,

$$\begin{aligned}
(H^{(i)}_{-i})^{-1}v^{(i)}_{-i} &= \left(G^- \odot (R^TR)_{-i}\right)^{-1} \cdot (1-p)pa^2(R^TR_{*i})_{-i} \\
&= \left(\frac{1}{(1-p)pa^2}G^- \odot (R^TR)_{-i}\right)^{-1}(R^TR_{*i})_{-i} \\
&= \frac{1}{1-p}\left((R^TR)_{-i} + \frac{p}{1-p}I \odot (R^TR)_{-i}\right)^{-1}(R^TR_{*i})_{-i}
\end{aligned} \tag{26}$$

Next, remember that $C^{-1} = R^TR + \frac{p}{1-p}I \odot R^TR$. Since no column of R is a zero vector, $I \odot R^TR$ is positive definite, thus C^{-1} is positive definite. By the properties of matrix inverse, for an invertible matrix E, if we swap the i-th and j-th rows (or columns) of E and get E', then E'^{-1} is equivalent to the matrix formed by swapping the i-th and j-th columns (or rows) of E^{-1}. Therefore, suppose $(C^{-1})^{\langle i \rangle}$ is obtained from C^{-1} by first swapping the kth row with the $(k+1)$th row for $k = i, i+1, ..., n-1$ sequentially, then swapping the kth column with the $(k+1)$th column for $k = i, i+1, ..., n-1$ sequentially. We have

$$(C^{-1})^{\langle i \rangle} = \begin{bmatrix} (R^TR)_{-i} + \frac{p}{1-p}I \odot (R^TR)_{-i} & (R^TR_{*i})_{-i} \\ (R^TR_{*i})^T_{-i} & \frac{1}{1-p}(R^TR)_{ii} \end{bmatrix}$$

and

$$\left((C^{-1})^{\langle i \rangle}\right)^{-1} = C^{\langle i \rangle} = \begin{bmatrix} M & C_{*i,-i} \\ C^T_{*i,-i} & C_{ii} \end{bmatrix}$$

where M is an $(n-1) \times (n-1)$ matrix that we are not interested in.

By the symmetric block matrix inverse (0.7.3, (Horn & Johnson, 2012)), we know that

$$\begin{bmatrix} A & B^T \\ B & D \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} + A^{-1}B^TS^{-1}BA^{-1} & -A^{-1}B^TS^{-1} \\ -S^{-1}BA^{-1} & S^{-1} \end{bmatrix}$$

where $S = D - BA^{-1}B^T$.

Let $A = (R^T R)_{-i} + \frac{p}{1-p}(I \odot (R^T R)_{-i})$, $B^T = (R^T R_{*i})_{-i}$ and $S^{-1} = C_{ii}$, we have

$$C_{*i,-i} = -A^{-1}B^T S^{-1} = -\left((R^T R)_{-i} + \frac{p}{1-p} I \odot (R^T R)_{-i}\right)^{-1} (R^T R_{*i})_{-i} \cdot C_{ii} \tag{27}$$

Combining Eq (27) and Eq (26), we get Eq (25), thereby completing the proof.

□

Proof of Proposition 3.5:

(a) Similar to Eq (11), we can expand objective function of Eq (14) as

$$l_{\mathcal{B}}(W) + \lambda\|W\|_F^2 = \sum_{i=1}^{n} h_{\mathcal{B}}^{(i)}(W_{*i}), \quad \text{where}$$
$$h_{\mathcal{B}}^{(i)}(W_{*i}) = W_{*i}^T H^{(i)} W_{*i} - 2W_{*i}^T v^{(i)} + \lambda\|W_{*i}\|_F^2 + \mathbb{E}_{\Delta\sim\mathcal{B}}\left[R_{*i}^T {A^{(i)}}^2 R_{*i}\right]$$

Hence, $\left[\frac{\partial h_{\mathcal{B}}^{(i)}}{\partial W_{*i}}\right]^T = 2H^{(i)}W_{*i} - 2v^{(i)} + 2\lambda W_{*i}$, and the solution of $\left[\frac{\partial h_{\mathcal{B}}^{(i)}}{\partial W_{*i}}\right]^T = 0$ becomes

$$W_{*i}^* = \left(H^{(i)} + \lambda I\right)^{-1} v^{(i)} \tag{28}$$

The optimal W^* is obtained by solving W_{*i}^* via Eq (28) for all i. The solution is unique since each $h_{\mathcal{B}}^{(i)}$ is strictly convex.

(b) Eq (15) is equivalent to solving for the stationary points of the following a Lagrangian function

$$\mathcal{L}(W,\, \mu) = l_{\mathcal{B}}(W) + \mu^T \text{diag}(W)$$

where $\mu \in \mathbb{R}^n$. Since

$$\mathcal{L}(W,\, \mu) = \sum_{i=1}^{n} \bar{h}_{\mathcal{B}}^{(i)}(W_{*i}, \mu_i), \quad \text{where}$$
$$\bar{h}_{\mathcal{B}}^{(i)}(W_{*i}, \mu_i) = W_{*i}^T H^{(i)} W_{*i} - 2W_{*i}^T v^{(i)} + \mu_i W_{ii} + \mathbb{E}_{\Delta\sim\mathcal{B}}\left[R_{*i}^T {A^{(i)}}^2 R_{*i}\right]$$

the solution (W, μ) of the system of equations

$$\left[\frac{\partial \mathcal{L}}{\partial W}\right]^T = \left[\left[\frac{\partial \mathcal{L}}{\partial W_{*1}}\right]^T, \left[\frac{\partial \mathcal{L}}{\partial W_{*2}}\right]^T, ..., \left[\frac{\partial \mathcal{L}}{\partial W_{*n}}\right]^T\right] = \left[\left[\frac{\partial \bar{h}_{\mathcal{B}}^{(1)}}{\partial W_{*1}}\right]^T, \left[\frac{\partial \bar{h}_{\mathcal{B}}^{(2)}}{\partial W_{*2}}\right]^T, ..., \left[\frac{\partial \bar{h}_{\mathcal{B}}^{(n)}}{\partial W_{*n}}\right]^T\right] = 0$$

$$\left[\frac{\partial \mathcal{L}}{\partial \mu}\right]^T = \left[\frac{\partial \mathcal{L}}{\partial \mu_1}, \frac{\partial \mathcal{L}}{\partial \mu_2}, ..., \frac{\partial \mathcal{L}}{\partial \mu_n}\right]^T = \left[\frac{\partial \bar{h}_{\mathcal{B}}^{(1)}}{\partial \mu_1}, \frac{\partial \bar{h}_{\mathcal{B}}^{(2)}}{\partial \mu_2}, ..., \frac{\partial \bar{h}_{\mathcal{B}}^{(n)}}{\partial \mu_n}\right]^T = 0$$

is given by taking

$$\left[\frac{\partial \bar{h}_{\mathcal{B}}^{(i)}}{\partial W_{*i}}\right]^T = 2H^{(i)}W_{*i} - 2v^{(i)} + \mu_i l^{(i)} = 0 \tag{29}$$
$$\frac{\partial \bar{h}_{\mathcal{B}}^{(i)}}{\partial \mu_i} = W_{ii} = 0 \tag{30}$$

for $i = 1, 2, ..., n$. Solving Eq (29), we get

$$W_{*i} = {H^{(i)}}^{-1}(v_i - \frac{1}{2}\mu_i l^{(i)}) \tag{31}$$

Combining Eq (30) and Eq (31), we have

$$W_{ii} = ({H^{(i)}}^{-1} v^{(i)})_i - \frac{1}{2}\mu_i ({H^{(i)}}^{-1} l^{(i)})_i = 0 \implies \mu_i = 2\frac{({H^{(i)}}^{-1} v^{(i)})_i}{({H^{(i)}}^{-1} l^{(i)})_i} \tag{32}$$

Finally, plugging Eq (33) into Eq (31), we get the solution of W^*: For any i,

$$W^*_{*i} = {H^{(i)}}^{-1}(v^{(i)} - \frac{({H^{(i)}}^{-1} v^{(i)})_i}{({H^{(i)}}^{-1} l^{(i)})_i} l^{(i)}) = {H^{(i)}}^{-1} v^{(i)} - \frac{({H^{(i)}}^{-1} v^{(i)})_i}{({H^{(i)}}^{-1} l^{(i)})_i} {H^{(i)}}^{-1} l^{(i)} \tag{33}$$

The solution Eq (33) is unique. By *second order sufficiency conditions* (Section 11.5, (Luenberger & Ye, 2008)), one can show that any W^* that minimizes $\mathcal{L}(W,\, \mu)$ is a strict local minimizer. Thus, the solution Eq (33) gives the global minimizer.

□

B Decoupled Expected Quadratic Loss with Low-Rank Constraint

This section discusses the closed-form solution of Eq (7) under low-rank constraint of W. Given the rank k ($k \leq n$), we would like to solve

$$\underset{W}{\text{argmin}}\ l_{\boldsymbol{\mathcal{D}}}(W) = \sum_{i=1}^{n} \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}} \left[\|Y_{*i} - XW_{*i}\|_F^2\right] \quad \text{s.t. rank}(W) \leq k \tag{34}$$

Theorem B.1. *Suppose $\mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}[X^T X]$ is independent of i, and denote*

$$\Sigma_{xx} = \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}[X^T X]$$
$$\Sigma_{xy} = \left[\mathbb{E}_{(X,Y)\sim\mathcal{D}^{(1)}}[X^T Y_{*1}], \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(2)}}[X^T Y_{*2}], ..., \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(n)}}[X^T Y_{*n}]\right]$$

If Σ_{xx} is non-singular, then the closed-form solution of Eq (34) is given by

$$W^* = \Sigma_{xx}^{-1/2} \left[\Sigma_{xx}^{-1/2}\Sigma_{xy}\right]_k \tag{35}$$

Here, let $\Sigma_{xx}^{-1/2}\Sigma_{xy} = U \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & ... & \\ & & & \sigma_n \end{bmatrix} V^T$ be the singular value decomposition where $\sigma_1 \geq \sigma_2 ... \geq \sigma_n$, we denote $\left[\Sigma_{xx}^{-1/2}\Sigma_{xy}\right]_k = \sum_{i=1}^{k} \sigma_i U_{*i} V_{*i}^T$.

Proof: Denote $y^{(i)} = \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}[Y_{*i}^T Y_{*i}]$. By Eq (7),

$$\begin{aligned}
l_{\mathcal{D}}(W) &= \sum_{i=1}^{n} W_{*i}^T \Sigma_{xx} W_{*i} - 2W_{*i}^T (\Sigma_{xy})_{*i} + y^{(i)} \\
&= \sum_{i=1}^{n} \left(\Sigma_{xx}^{1/2} W_{*i}\right)^T \Sigma_{xx}^{1/2} W_{*i} - 2\left(\Sigma_{xx}^{1/2} W_{*i}\right)^T \Sigma_{xx}^{-1/2} (\Sigma_{xy})_{*i} + y^{(i)} \\
&= \sum_{i=1}^{n} \left\|\Sigma_{xx}^{1/2} W_{*i} - \Sigma_{xx}^{-1/2} (\Sigma_{xy})_{*i}\right\|_F^2 + y^{(i)} - (\Sigma_{xy})_{*i}^T \Sigma_{xx}^{-1} (\Sigma_{xy})_{*i} \\
&= \left\|\Sigma_{xx}^{1/2} W - \Sigma_{xx}^{-1/2}\Sigma_{xy}\right\|_F^2 + \sum_{i=1}^{n} y^{(i)} - (\Sigma_{xy})_{*i}^T \Sigma_{xx}^{-1} (\Sigma_{xy})_{*i}
\end{aligned}$$

By Eckart–Young–Mirsky theorem, $\left[\Sigma_{xx}^{-1/2}\Sigma_{xy}\right]_k = \underset{\text{rank}(Q)\leq k}{\text{argmin}} \left\|Q - \Sigma_{xx}^{-1/2}\Sigma_{xy}\right\|_F^2$ for any $Q \in \mathbb{R}^{n\times n}$. Therefore, $\Sigma_{xx}^{1/2} W^* = \left[\Sigma_{xx}^{-1/2}\Sigma_{xy}\right]_k \implies W^* = \Sigma_{xx}^{-1/2}\left[\Sigma_{xx}^{-1/2}\Sigma_{xy}\right]_k$. □

Note that the low-rank solution Eq (35) is not applicable when $\Sigma_{xx} = \mathbb{E}_{(X,Y)\sim\mathcal{D}^{(i)}}[X^T X]$ depends on i: If Σ_{xx} varies with i, then in the proof $\sum_{i=1}^{n} \left\| \Sigma_{xx}^{1/2} W_{*i} - \Sigma_{xx}^{-1/2} \left(\Sigma_{xy}\right)_{*i} \right\|_F^2$ cannot be combined into $\left\| \Sigma_{xx}^{1/2} W - \Sigma_{xx}^{-1/2} \Sigma_{xy} \right\|_F^2$.

The low-rank solution Eq (35) is applicable to EDLAE only when taking $a = b$: by Eq (10), Σ_{xx} is represented by $H^{(i)}$; by Lemma 3.2, if $a = b$, $H^{(i)}$ will have all diagonal entries being $(1-p)b^2$ and all off-diagonal entries being $(1-p)^2 b^2$ for any i, thus being independent of i. However, when $a \neq b$, $H^{(i)}$ will depend on i, making Eq (34) not applicable.

C Related Works

The evolution of collaborative filtering (CF) in recommendation systems has undergone several key paradigm shifts. In its early stages, neighborhood-based methods dominated the field, with influential works such as user-item KNN approaches (Hu et al., 2008) and sparse linear models (SLIM) (Ning & Karypis, 2011) setting the foundation. However, the Netflix Prize competition marked a turning point, accelerating the adoption of matrix factorization (MF) techniques, which offered improved scalability and latent feature learning (Koren et al., 2009). These models aim to solve a matrix completion problem, which has been extensively studied theoretically (Candès & Tao, 2010; Recht, 2011; Foygel et al., 2011; Shamir & Shalev-Shwartz, 2011).

The rise of deep learning (LeCun et al., 2015) further revolutionized the landscape, introducing more expressive neural architectures. Among these, graph-based models gained prominence, including Neural Collaborative Filtering (NCF) (He et al., 2017), which replaced traditional MF with neural networks, and later refinements like Neural Graph Collaborative Filtering (NGCF) (Wang et al., 2019) and LightGCN (He et al., 2020), which explicitly leveraged graph structures for higher-order user-item relationship modeling. Simultaneously, industry-scale solutions emerged, blending memorization and generalization through hybrid architectures such as Wide & Deep (Cheng et al., 2016), DeepFM (Guo et al., 2017), and Deep & Cross Networks (DCN) (Wang et al., 2017), which automated feature interactions while maintaining interpretability.

Alongside the ongoing research that explores various models to enhance recommendation performance, the research community has gradually recognized the importance of gaining a deeper theoretical understanding of loss functions (Terven et al., 2025; Wu et al., 2024b). These theoretical investigations seek to reveal the fundamental principles and mathematical underpinnings that govern the behavior and optimization direction of recommendation systems, thereby advancing our overall understanding of how these systems function. BPR (Rendle et al., 2009). Early methods often adopted pointwise L2 loss over observed ratings or implicit feedback (Hu et al., 2008), which is simple and analytically tractable. Later, pairwise ranking losses such as BPR (Rendle et al., 2009) became popular for top-N recommendation, optimizing relative preferences between positive and negative items. Softmax-based listwise losses, such as sampled softmax (Jannach et al., 2010) were introduced to better align with ranking metrics like NDCG. In recent years, contrastive learning frameworks (Zhou et al., 2021; Wang et al., 2022) have gained prominence as a powerful and effective approach, particularly in unsupervised recommendation scenarios. Notable studies such as (Li et al., 2023; Liu et al., 2021) have further showcased their effectiveness in this domain.

Notably, recent studies (Rendle, 2010) have shown that well-tuned linear models can outperform more complex deep architectures on sparse implicit data, challenging the assumption that greater model expressiveness always yields better performance. At the same time, alternative objectives such as pairwise ranking losses (Rendle et al., 2009), listwise softmax, and contrastive formulations (Zhou et al., 2021; Wang & Liu, 2021)—though popular—often suffer from instability, sensitivity to negative sampling, and increased computational overhead. Motivated by these findings, we adopt a different perspective: rather than seeking more complex losses or models, we focus on refining linear models under the classical L2 loss.

LAEs are one type of the linear recommender models. One of the earliest LAE model is SLIM (Ning & Karypis, 2011), which trains the loss $\|R - RW\|_F^2$ with L1 and L2 regularizers. The zero-diagonal constraint was first introduced in EASE (Steck, 2019) to prevent solutions from overfitting toward the identity matrix. EDLAE (Steck, 2020) instead employs dropout and emphasis as an alternative

strategy to mitigate overfitting. ELSA (Vančura et al., 2022) construct the model W with a zero diagonal by enforcing $W = AA^T - I$ for some matrix A subject to $\|A_{i*}\|_2^2 = 1$ for all i. (Moon et al., 2023) shows that a strict zero-diagonal constraint does not always yield the best performance, and that replacing it with a diagonal bounded by a small norm during training can improve results.

D Supplemental Experiments on Time and Memory Cost

In this section, we provided more experimental details and results (as in Table 3) regarding training time and memory usage.

Table 3 compares the time and memory costs of deep learning models and LAEs. It shows that for large datasets like Yelp 2018, DEQL families finish training in just 8 minutes on a CPU, which is much faster than all other advanced deep learning baselines.

The primary reason LAE-based methods (e.g., DEQL) exhibit higher memory cost but lower training time than deep learning-based methods is due to their computational paradigm: deep learning-based models train via batch gradient descent, loading only small batches into GPU memory at each step, which keeps memory usage around 3GB but requires many iterations, leading to longer training time. In contrast, LAE-based methods load the entire matrix into memory to compute its inverse as part of a closed-form solution, which may require 80GB for datasets like Yelp2018. This space-time trade-off enables the entire training process to finish much faster, as shown in our experiments.

Unlike GPU-intensive GNN-based models such as LightGCN or SSM (GNN), DEQL families are particularly suitable for deployment in memory-limited or CPU-only environments, making it highly practical. In practice, modern CPU servers often equip with 500 GB – 1 TB RAM, mitigating this issue.

Table 3: Approximate Training time and memory usage on Yelp2018. Deep based models mainly consume GPU memory, while others rely on CPU memory.

Model	Time (min)	Memory (GB)	Memory Type
LightGCN	30	1	GPU
SimpleX (GNN)	130	1	GPU
SSM (MF)	13	1	GPU
SSM (GNN)	13	1	GPU
DLAE	2	70	CPU
EASE	2	60	CPU
EDLAE	4	70	CPU
DEQL	6	80	CPU
DEQL (L2+zero-diag)	8	80	CPU
DEQL (L2)	8	80	CPU

E Discussions

E.1 Limitations

One limitation of our work is that DEQL is currently used only as an optimization tool, providing closed-form solutions under given hyperparameters. However, it does not offer guidance on which hyperparameter choices lead to improved testing performance. As a result, hyperparameter selection still relies on empirical tuning, and its theoretical understanding remains underexplored.

Another limitation lies in the computational complexity of the Algorithm in Section 4. Even when R is sparse, this algorithm still has an $O(n^3)$ complexity bottleneck due to the need to compute the inverse of H_0. By Theorem 3.3, H_0 is positive definite if no columns of R is a zero vector. In this case, the inverse of H_0 is typically performed by Cholesky decomposition (Krishnamoorthy & Menon, 2013), but it still costs $O(n^3)$.

E.2 Explanation for the Performance Gains from Regularization

Here we provide a possible explanation for the experimental results in Table 1 and Table 2, which show why the performance of DEQL(L2) and DEQL(L2 + zero-diag) surpasses that of plain DEQL. According to statistical learning theory (Vapnik, 1999), searching for solutions within a large hypothesis space often leads to overfitting, while searching in a small hypothesis space may cause underfitting – both cases resulting in poor testing performance. Structural risk minimization (SRM) (Vapnik, 1999) addresses this trade-off by controlling the size of the hypothesis space. In this context, both the L2 regularizer and the zero-diagonal constraint can be interpreted as SRM techniques that restrict the hypothesis space. Let $\mathcal{U}_{\text{DEQL}}, \mathcal{U}_{\text{DEQL(L2)}}$ and $\mathcal{U}_{\text{DEQL(L2 + zero-diag)}}$ denote the hypothesis spaces of DEQL, DEQL(L2) and DEQL(L2 + zero-diag), respectively. Then we have the nested relationship $\mathcal{U}_{\text{DEQL(L2 + zero-diag)}} \subset \mathcal{U}_{\text{DEQL(L2)}} \subset \mathcal{U}_{\text{DEQL}}$. However, the hypothesis space that yields the best performance depends on the dataset, as reflected in the results: on some datasets DEQL(L2 + zero-diag) performs better, while on others DEQL(L2) performs better.

LLM Usage Statement

We use ChatGPT solely for polishing writing at the sentence and paragraph level. The content and contributions of this paper were created by the authors. All text refined with ChatGPT has been carefully checked to avoid errors.