Electronic Research Archive

https://www.aimspress.com/journal/era

ERA, 34(5): 2947–2973.
DOI: 10.3934/era.2026134
Received: 27 January 2026
Revised: 24 March 2026
Accepted: 27 March 2026
Published: 03 April 2026

Research article

Learning nonparanormal concept graphs from language representations

Jizheng Lai and Jianxin Yin*

Center for Applied Statistics and School of Statistics, Renmin University of China, Beijing 100872, China

* **Correspondence:** Email: jyin@ruc.edu.cn.

Abstract: Learning conditional dependence structure among latent concepts from unstructured text can provide an interpretable graph prior for downstream models. Existing methods estimate such a concept graph by modeling GloVe-style concept embeddings as matrix-normal observations with sparse row/column precision matrices, but its Gaussian assumption is usually not satisfied for modern neural embeddings. In this paper, we extend this framework to a nonparanormal matrix-normal graphical model, allowing monotone marginal transformations while preserving the conditional-independence interpretation of the row/column precision matrix. We construct matrix-valued concept embeddings from language model representations and estimate sparse row/column precision matrices via a penalized likelihood that replaces sample covariances with rank-based correlation estimates, improving the robustness in cases with skewness and heavy tails. The resulting concept graph is defined by the support of the row precision matrix and is used as a structural prior for downstream prediction. Experiments on a salary-prediction task show that the proposed nonparanormal precision graph consistently reduces absolute-error risk and improves stability over competitors. Ablations further indicate that performance gains stem from meaningful conditional-dependence structure, rather than merely from increased model capacity.

Keywords: concept graph; graphical neural network; language representation; matrix normal graphical model; nonparanormal transformation

1. Introduction

Pretrained language models (LMs) have become a default tool for turning unstructured text into high-dimensional representations, powering a wide range of downstream analyses and decision-support tasks. Recent instruction-tuned and open foundation models (e.g., InstructGPT, LLaMA, and Qwen series) further strengthen the semantic fidelity of such representations and make it practical to extract concept-level signals at scale [1–4].

Despite their empirical success, most LM-based pipelines remain "vector-first": Concepts are embedded as high-dimensional vectors, while the dependence structure among concepts is left implicit. In many scientific and policy settings, however, the key question is not only how to represent concepts, but also which concepts remain directly related after conditioning on the others. Such conditional-dependence information naturally supports transparent scientific hypotheses, reusable structural priors, and more stable downstream learning [5, 6].

Graphical modeling offers a principled language to represent conditional independence through sparsity in precision matrices. However, directly applying Gaussian graphical models to modern LM embeddings is statistically questionable, because neural representations are often far from Gaussian distributed and may exhibit anisotropy or heavy-tailed behavior [7–9]. A standard remedy is to move from a fully parametric Gaussian assumption to nonparanormal modeling, which preserves the conditional-independence interpretation while allowing unknown monotone marginal distortions [10]. Recent work has further extended rank-based Gaussian-copula ideas to more structured settings, underscoring the practical value of semiparametric dependence estimation beyond the Gaussian case [11].

A recent step toward concept-graph learning from text representations is a matrix-normal framework that models matrix-valued concept embeddings with sparse row/column precision matrices to recover an interpretable conditional-dependence concept graph [12]. However, that framework is tied to a Gaussian assumption and empirically focuses on GloVe-style embeddings that are closer to Gaussianity. More broadly, semiparametric graph-estimation methods provide robustness beyond Gaussian assumptions [10, 11, 13, 14], while graph-based downstream learning methods, including graph neural network (GNN)-based and multiview graph approaches, demonstrate the practical value of incorporating structured relational information into prediction [15–18]. Related ideas on explicit structure learning and regularization have also proved useful in other structured prediction settings, such as spatiotemporal tensor modeling with missing data [19]. Nevertheless, these lines of work have not been integrated to estimate an interpretable conditional-dependence concept graph from modern LM-derived embeddings under a matrix-normal framework. This leaves open a central methodological question: Can we perform principled conditional-dependence graph estimation for concept embeddings produced by modern LMs, where Gaussianity is typically violated?

In this paper, we address this question by extending the Matrix-GloVe framework to a nonparanormal matrix-normal setting. Concretely, we treat each corpus (and its bootstrap variants) as producing a matrix-valued observation $Y \in \mathbb{R}^{p\times q}$, where each row corresponds to a concept and each column corresponds to an embedding dimension. Under a nonparanormal graphical model, we retain the conditional-dependence interpretation of the row precision matrix A while gaining robustness to skewness and heavy tails via rank-based dependence estimation. This yields an interpretable graph prior that can be injected into downstream predictors and evaluated through predictive risk and stability.

The main contributions of this work are listed below:

- We extend matrix-normal concept graph learning from the Gaussian setting to a nonparanormal framework, enabling conditional-dependence graph estimation from modern LM embeddings under weaker distributional assumptions.
- We develop a scalable rank-based plug-in estimator for matrix-valued LM concept embeddings and integrate it into penalized precision learning. Rather than operating on the $pq \times pq$

dimensional $\widehat{R}$ in (2.2), we decompose the computation through two reduced matrices (R_A, R_B) of sizes $p \times p$ and $q \times q$, substantially improving computational and memory efficiency for large p and q.

- We evaluate the learned concept graph not only by predictive performance but also by stability, measured by variability Std(MSE). We further include a matched-sparsity random-graph control, that differs only in the adjacency used for message passing, to verify that the gains are attributable to the learned conditional-dependence structure.

2. Preliminaries

To make the paper self-contained, this section briefly introduces the main methodological background underlying our framework. We review the matrix normal graphical model, its nonparanormal extension for handling non-Gaussian data, and the graph neural network machinery used in the downstream prediction stage.

2.1. Matrix normal graphical models

We briefly review matrix normal graphical models (MNGM) [20], as they provide the statistical foundation for our framework. The presentation here is standard and included only to make the paper self-contained.

Definition 2.1 (Matrix normal distribution)**.** Assume $Y \in \mathbb{R}^{p\times q}$ is a matrix-valued random variable. We say Y follows a matrix normal distribution, denoted by $Y \sim MN_{p,q}(M; U, V)$, if it has density

$$p(Y \mid M, U, V) = (2\pi)^{-pq/2}|U|^{-q/2}|V|^{-p/2}\exp\left(-\frac{1}{2}\operatorname{tr}[(Y-M)^\top U^{-1}(Y-M)V^{-1}]\right),$$

where M is the mean matrix, and U and V are the row and column covariance matrices, respectively. Equivalently, $\operatorname{vec}(Y) \sim N_{pq}(\operatorname{vec}(M),\ V\otimes U)$. We write the corresponding precision matrices as $A = U^{-1}$ and $B = V^{-1}$.

A key property of MNGM is that sparsity in the precision matrices $A = U^{-1}$ and $B = V^{-1}$ admits a conditional-independence interpretation [20]: For $\gamma \neq \mu$, $b_{\gamma\mu} = 0$ implies that the γ-th and μ-th columns of Y are conditionally independent given the remaining columns; similarly, for $\delta \neq \eta$, $a_{\delta\eta} = 0$ implies conditional independence between the δ-th and η-th rows given the remaining rows. In our setting, rows of Y index concepts and columns index embedding dimensions; therefore, the concept conditional-dependence graph is induced by the *off-diagonal* support of the row precision matrix A. For notational simplicity, we work with centered observations and reuse Y_k to denote $Y_k - \bar{Y}$, so we set $M = 0$ in what follows.

In practice, A and B are typically estimated by penalized likelihood to encourage sparsity (e.g., ℓ_1 penalties). Given matrix observations $\{Y_k\}_{k=1}^n$, one common formulation minimizes the penalized negative log-likelihood:

$$\phi(A, B) = -q\log|A| - p\log|B| + \frac{1}{n}\sum_{k=1}^{n}\operatorname{tr}(AY_kBY_k^\top) + \lambda\sum_{i\neq j}|a_{ij}| + \rho\sum_{i\neq j}|b_{ij}|, \tag{2.1}$$

where λ and ρ control the sparsity levels of A and B, respectively (e.g., via an edge budget or validation).

2.2. Nonparanormal MNGM

To accommodate non-Gaussian LM embeddings, we adopt a nonparanormal matrix normal graphical model [13], which assumes that the matrix-valued observations admit a structured semiparametric approximation: There exist entry-wise strictly monotone transformations $f = \{f_{ij}\}$ such that

$$\mathrm{vec}(f(Y)) \sim N_{pq}(\mathrm{vec}(M),\ V \otimes U).$$

Here, the monotone transformation relaxes marginal Gaussianity, whereas the Kronecker-separable covariance $V \otimes U$ is an additional modeling assumption used to parsimoniously encode row-wise and column-wise dependence. We do not claim that entry-wise monotonicity alone implies matrix normality; rather, this assumption defines the model family adopted in this work. This approximation is appealing in our setting because concept embeddings are naturally organized as matrices with concept and embedding-dimension axes, and the separable covariance form provides a tractable way to recover the row precision structure that defines the concept graph while treating column dependence as a nuisance component.

Instead of estimating f, we directly estimate the latent correlation matrix of $\mathrm{vec}(f(Y))$ using rank-based plug-in estimators. Given samples $\{Y_k\}_{k=1}^n$, let $x_k = \mathrm{vec}(Y_k) \in \mathbb{R}^{pq}$. In our implementation, the sample size n denotes the total number of matrix observations, and we index these observations by $k = 1, \ldots, n$. For each pair of coordinates (s, u) of x where $s \neq u$, we compute the sample rank correlation (Spearman's $\hat{\rho}_{su}$ or Kendall's $\hat{\tau}_{su}$ [13]) and set

$$\widehat{R}_{su} = \begin{cases} 2\sin\left(\frac{\pi}{6}\hat{\rho}_{su}\right), & \text{(Spearman's } \rho\text{)} \\ \sin\left(\frac{\pi}{2}\hat{\tau}_{su}\right), & \text{(Kendall's } \tau\text{)}. \end{cases}$$

We then replace the sample covariance term in (2.1) by the rank-based plug-in matrix $\widehat{R}$, yielding the penalized objective

$$\phi(A, B) = -q \log|A| - p \log|B| + \mathrm{tr}\left((B \otimes A)\widehat{R}\right) + \sum_{i \neq j} p_\lambda(a_{ij}) + \sum_{i \neq j} p_\rho(b_{ij}), \tag{2.2}$$

where p_λ and p_ρ are sparsity-inducing penalties (we use ℓ_1 in experiments). Optimization follows the same alternating/iterative scheme as the Gaussian case, with $\widehat{R}$ in place of the sample covariance, preserving computational efficiency while improving robustness to skewness and heavy tails. Beyond the classical nonparanormal graph estimation literature, recent semiparametric Gaussian-copula graphical model work also considers richer observation structures (e.g., multi-attribute measurements) while retaining interpretable conditional-dependence graphs [14].

2.3. Graph neural networks

Graph neural networks (GNNs) have emerged as a powerful class of models for learning representations from graph-structured data. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with node features $\{x_v\}_{v \in \mathcal{V}}$, GNNs aim to learn node, edge, or graph-level embeddings by iteratively aggregating information from local neighborhoods. A general message-passing framework can be written as

$$h_v^{(l+1)} = \phi^{(l)}\left(h_v^{(l)},\ \mathrm{AGG}\left(\{\psi^{(l)}(h_v^{(l)}, h_u^{(l)}, e_{uv}) : u \in \mathcal{N}(v)\}\right)\right), \tag{2.3}$$

where $h_v^{(l)}$ denotes the representation of node v at layer l, $\mathcal{N}(v)$ is the neighborhood of v, AGG$(\cdot)$ is a permutation-invariant aggregation operator, $\phi^{(l)}$ and $\psi^{(l)}$ are learnable functions, and e_{uv} represents edge attributes, which are omitted in our implementation.

Early GNN variants, such as graph convolutional networks (GCNs) [15] and GraphSAGE [16], instantiate this framework using linear transformations and mean or sum aggregation, while attention-based models, such as graph attention networks (GATs) [17], introduce adaptive importance weights over neighbors. These models have been successfully applied to a wide range of tasks, including node classification [21], fraud detection [22], and 3D segmentation [23].

From a modeling perspective, GNNs use the input graph as a carrier of relational information and propagate neighborhood signals through message passing. In our setting, the graph is not treated as an arbitrary similarity graph; instead, it is supplied by the estimated concept conditional-dependence structure from the graphical model. The resulting graph therefore serves as a structured prior for downstream representation learning.

3. Methods

We propose the nonparanormal graph via language model (npGraph-LM), a statistically grounded approach to recover concept-level conditional dependence structure from text representations produced by modern language models. The target is not merely to improve prediction, but to obtain a sparse and interpretable concept graph that can serve as a reusable structural prior. The approach proceeds through paragraph bootstrap for stabilizing rank-based dependence estimation, matrix-valued concept embedding construction, nonparanormal matrix graphical modeling to estimate sparse precision structure, and finally document-specific concept-induced subgraph extraction whose graph representations are concatenated with text representations for downstream prediction. An overview of the npGraph-LM pipeline is shown in Figure 1.

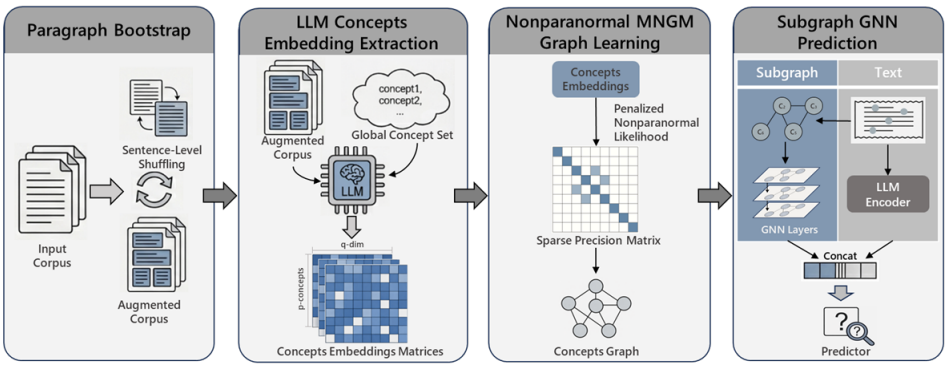


Figure 1. Overview of npGraph-LM. We generate multiple augmented views of each document via paragraph bootstrap, encode concepts into matrix-valued embeddings, build a sparse concept conditional-dependence graph using a nonparanormal matrix graphical model (via rank-based dependence estimation and penalized precision learning), and extract document-specific concept-induced subgraphs, concatenate the resulting subgraph representation with the text representation, and perform downstream prediction.

3.1. Paragraph bootstrap for corpus augmentation

Following [12], we use *paragraph bootstrap* to create T corpus-level augmented views. Let the corpus be $\mathcal{D} = \{d_i\}_{i=1}^{N_{\text{doc}}}$. At bootstrap round t, we apply paragraph-level sentence shuffling within each document d_i to obtain a bootstrapped corpus:

$$\widetilde{\mathcal{D}}^{(t)} = \{\tilde{d}_i^{(t)}\}_{i=1}^{N_{\text{doc}}}, \qquad t = 1, \ldots, T.$$

We then encode the entire bootstrapped corpus $\widetilde{\mathcal{D}}^{(t)}$ to construct a single matrix-valued concept embedding

$$Y^{(t)} \in \mathbb{R}^{p \times q},$$

whose rows correspond to concepts in the global vocabulary $C = \{c_1, \ldots, c_p\}$ and columns correspond to embedding dimensions. Therefore, $\{Y^{(t)}\}_{t=1}^{T}$ constitutes $n := T$ matrix observations for subsequent dependence estimation.

After training/encoding on these bootstrap corpus variants, we obtain multiple realizations of the concept-embedding matrix, effectively increasing the sample size for dependence estimation and stabilizing subsequent rank-based correlation corpus estimates without introducing parametric assumptions.

Because modern LMs are pretrained on large-scale heterogeneous corpora and our target is concept-level representation rather than discourse generation, we expect the resulting embeddings to be relatively robust to such mild paragraph-local reordering. Empirical evidence on embedding stability under paragraph bootstrap is provided in Appendix A.4.

3.2. Concept extraction and embedding tensor construction

For each augmented document $\tilde{d}_i^{(t)}$, we employ a large language model (LLM) to identify a set of salient concepts $C = \{c_1, \ldots, c_p\}$ shared across the corpus. Each concept c_j is encoded into a q-dimensional embedding vector using the LLM after unsupervised training on these bootstrap document variants. Collecting all concept embeddings yields a matrix-valued observation:

$$Y^{(t)} \in \mathbb{R}^{p \times q},$$

where rows correspond to concepts and columns correspond to embedding dimensions.

The collection $\{Y^{(t)}\}$ can thus be viewed as samples of a matrix-valued random variable, naturally aligning with the matrix normal and nonparanormal graphical modeling framework introduced in Section 2. In practice, we construct a global concept vocabulary by aggregating concept candidates across the corpus, standardizing synonymous expressions, and aligning each document to the resulting vocabulary when forming $Y^{(t)}$.

When a concept set is not prespecified by the corpus, we construct a global concept vocabulary in three stages: concept candidate generation, concept standardization and synonym merging, and final vocabulary formation.

We collect candidate concepts from three complementary sources.

First, we extract textbook index terms from the statistics, data science, computer science, biostatistics, and economics textbooks included in the corpus. These index entries provide relatively curated domain concepts.

Second, we collect skill keywords explicitly appearing in job postings. Many recruitment texts list required skills, software tools, or knowledge areas in a structured or semi-structured form, and these keywords are also treated as concept candidates.

Third, to capture additional concept expressions appearing in free text, we use Qwen2 to extract concept-like keywords from documents. The model is prompted to identify short noun phrases corresponding to knowledge points or professional skills. A typical prompt template is as follows:

```
You are a professional information extraction assistant.

Your tasks are:
1) Identify the most appropriate category of the text
   (choose exactly one from a predefined list).
2) Extract several keywords or key phrases that represent
   knowledge concepts or skills appearing in the text.

Requirements:
- Keywords should be concise (preferably 2-10 characters in Chinese).
- Prefer noun phrases that represent concepts or skills.
- Avoid stop words, filler words, or informal expressions.
- Avoid duplicate terms.
- The output must be strictly formatted as JSON.
- Do not output any explanations or additional text.
```

These constraints help the model produce standard concept-level phrases instead of arbitrary text spans.

After collecting candidate concepts from the above sources, we perform deduplication and synonym merging with an LLM-assisted procedure. We use Qwen2 to judge whether two candidate terms refer to the same concept or skill.

For each candidate pair (a, b), the model decides whether they should be merged. If so, it outputs a canonical concept name. The prompt used in this step is:

```
You are a terminology normalization assistant.

Given a pair of terms (a, b), determine whether they represent the same
concept or skill.

If they should be merged into the same concept cluster, output the pair
and a canonical concept name.

If they should NOT be merged, output nothing.

Requirements:
- The canonical name should be a more standard or widely used term.
- Prefer concise Chinese terminology when possible.
- Keep widely used English abbreviations if appropriate.
```

```
- The output must be strictly formatted as JSON.
- Do not provide explanations.
```

The resulting canonical terms are used to merge synonymous concepts and remove duplicates, so that concept mentions with similar semantics are mapped consistently to the same global concept entry.

After synonym merging and deduplication, the remaining canonical concepts form the final vocabulary C. Each document is then aligned to this global vocabulary, and the matched concept indices are used in subsequent matrix-valued embedding construction and downstream document-specific concept activation.

3.3. Concept graph estimation via nonparanormal matrix normal graphical models

To capture conditional dependency relationships among concepts while allowing for non-Gaussian embedding distributions, we adopt a nonparanormal matrix normal graphical model. We assume that there exist monotonic transformations such that $\mathrm{vec}(f(Y)) \sim N_{pq}(\mathrm{vec}(M), V \otimes U)$, where $U \in \mathbb{R}^{p\times p}$ and $V \in \mathbb{R}^{q\times q}$ are the row and column covariance matrices, respectively.

Instead of explicitly estimating the transformation functions, we employ rank-based estimators to obtain a plug-in estimate $\widehat{R}$ of the latent correlation structure of $\mathrm{vec}(f(Y))$. The corresponding row and column precision matrices $A = U^{-1}$ and $B = V^{-1}$ are obtained by minimizing the penalized objective in (2.2). The sparsity pattern of the row precision matrix A defines a concept-level conditional independence graph $\mathcal{G}_c = (C, \mathcal{E}_c)$, where an edge indicates direct dependency between two concepts.

This statistically-grounded graph provides an interpretable and robust representation of concept relationships, which is subsequently used as structured prior knowledge for neural message passing.

Proposition 3.1 (Concept dependency interpretation)**.** Assume that the matrix-valued concept embedding $Y \in \mathbb{R}^{p\times q}$ follows a nonparanormal matrix graphical model, i.e., there exist entry-wise strictly monotone transformations $f = \{f_{ij}\}$ such that $\mathrm{vec}(f(Y)) \sim N_{pq}(\mathrm{vec}(M), V \otimes U)$. Let $A = U^{-1}$ be the row precision matrix. Let $\Delta = \{1, \ldots, p\}$ be the row index set. Then, for any two concepts c_δ and c_η (rows $\delta \neq \eta$),

$$a_{\delta\eta} = 0 \quad \Longleftrightarrow \quad Y^\delta \perp\!\!\!\perp Y^\eta \mid Y^{\Delta\setminus\{\delta,\eta\}},$$

and hence the support of A encodes the concept conditional-dependence graph.

Intuition. The nonparanormal model assumes that an entry-wise strictly monotone transformation f maps Y to a matrix-normal variable $\widetilde{Y} = f(Y)$. Since such coordinate-wise bijections do not mix entries, conditional independence relations among row-blocks are preserved under f. For the matrix-normal $\widetilde{Y} \sim MN_{p,q}(M; U, V)$, zeros in the row precision $A = U^{-1}$ correspond exactly to conditional independences between pairs of rows of $\widetilde{Y}$. Combining these two facts yields the claim that the support of A encodes the concept conditional-dependence graph.

Proof. See Appendix A.1.

The full graph estimation process is illustrated in Figure 2.

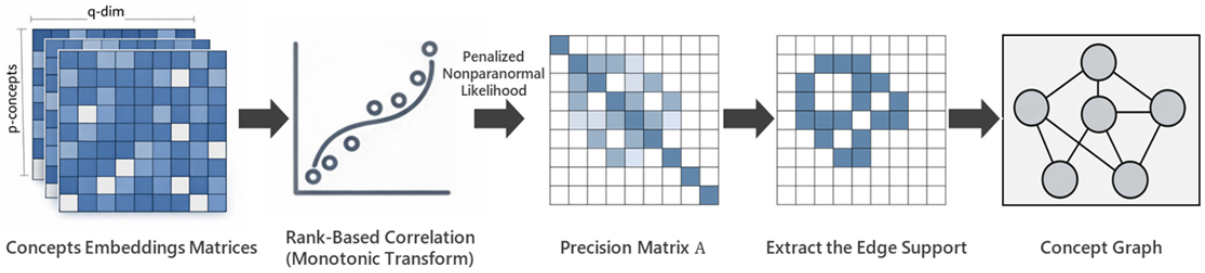


Figure 2. Concept graph estimation. Given the matrix-valued concept embeddings, we apply the nonparanormal graphical model to obtain a robust correlation structure and then fit a sparse row precision matrix $A = U^{-1}$. Nonzero off-diagonal entries in A indicate conditional dependencies between concepts, which are converted into the concept graph $\mathcal{G}_c$ used by the subsequent GNN encoder.

3.4. Rank-based plug-in construction of $\widehat{R}$ and efficient parallel computation

To handle unknown entry-wise strictly monotone distortions, we construct a rank-based plug-in estimate of the latent correlation structure and avoid explicit estimation of the marginal transformations. Following the semiparametric bigraphical-model treatment of [13, Proposition 3.2], although the conceptual target is a $pq \times pq$ plug-in matrix $\widehat{R}$ for $\mathrm{vec}(f(Y))$, the matrix-normal parameterization implies that the objective depends on $\widehat{R}$ only through two reduced matrices, denoted $R_A \in \mathbb{R}^{p\times p}$ and $R_B \in \mathbb{R}^{q\times q}$, which can be formed by structured aggregation of entries of $\widehat{R}$ with respect to the current (A, B) iterates.

Let $\{Y^{(t)}\}_{t=1}^{T}$ be matrix observations with $Y^{(t)} \in \mathbb{R}^{p\times q}$, and define $x^{(t)} = \mathrm{vec}(Y^{(t)}) \in \mathbb{R}^{pq}$. For each coordinate pair $(s, u) \in \{1, \ldots, pq\}^2$, we need to compute the sample Spearman rank correlation $\hat{\rho}_{su}$ and apply the standard sine calibration:

$$\widehat{R}_{su} = 2\sin\left(\frac{\pi}{6}\hat{\rho}_{su}\right). \tag{3.1}$$

This plug-in construction preserves robustness induced by ranking, while the sine mapping calibrates the rank correlation onto a correlation scale suitable for subsequent precision estimation.

We then construct two reduced matrices from $\widehat{R}$. Following [13, Proposition 3.2], define the selection matrices $\{K_\ell\}_{\ell=1}^{p}$ with $K_\ell \in \mathbb{R}^{pq\times q}$ whose $(\ell + p(j-1), j)$-th entry equals 1, and 0 otherwise $(j = 1, \ldots, q)$, so that K_ℓ extracts the q coordinates corresponding to the ℓ-th row of Y under $x = \mathrm{vec}(Y)$. Then, given the current iterate $\widehat{B}^{(k)}$, the (ℓ, m)-th entry of the reduced matrix $\widehat{R}_A^{(k)}$ can be written as

$$[\widehat{R}_A^{(k)}]_{\ell m} = \mathrm{tr}\left(K_\ell \widehat{B}^{(k)} K_m^\top \widehat{R}\right) = \mathrm{tr}\Big(\underbrace{K_m^\top \widehat{R} K_\ell}_{\text{a } q\times q \text{ block of } \widehat{R}} \quad \widehat{B}^{(k)}\Big), \qquad \ell, m \in \{1, \ldots, p\}, \tag{3.2}$$

where the second equality follows from cyclicity of the trace. Importantly, $K_m^\top \widehat{R} K_\ell \in \mathbb{R}^{q\times q}$ is exactly the sub-block of $\widehat{R}$ indexed by the coordinates $\{(m, 1), \ldots, (m, q)\}$ (rows) and $\{(\ell, 1), \ldots, (\ell, q)\}$ (columns), i.e., $[K_m^\top \widehat{R} K_\ell]_{j_1 j_2} = \widehat{R}_{(m,j_1),(\ell,j_2)}$. Thus, each entry of $\widehat{R}_A^{(k)}$ can be obtained by extracting this $q \times q$ block from $\widehat{R}$ and taking a weighted trace against $\widehat{B}^{(k)}$, without performing the explicit matrix multiplication on the full $pq \times pq$ matrix.

Similarly, let $\{L_r\}_{r=1}^{q}$ with $L_r \in \mathbb{R}^{p\times pq}$ denote the selection matrices whose r-th $p \times p$ block equals I_p, and 0 otherwise, so that L_r extracts the p coordinates corresponding to the r-th column of Y in $\mathrm{vec}(Y)$.

Given $\widehat{A}^{(k+1)}$, define $\widehat{R}_B^{(k+1)}$ entry-wise by

$$[\widehat{R}_B^{(k+1)}]_{rs} = \mathrm{tr}\Big(L_r^\top \widehat{A}^{(k+1)} L_s \widehat{R}\Big) = \mathrm{tr}\Big(\underbrace{L_s \widehat{R} L_r^\top}_{\text{a } p\times p \text{ block of } \widehat{R}} \widehat{A}^{(k+1)}\Big), \qquad r, s \in \{1, \dots, q\}, \tag{3.3}$$

again by cyclicity. Here $L_s \widehat{R} L_r^\top \in \mathbb{R}^{p\times p}$ is the sub-block of $\widehat{R}$ indexed by column groups s and r under the vectorization order. Consequently, each alternating update depends on $\widehat{R}$ only through the two reduced matrices $(\widehat{R}_A^{(k)}, \widehat{R}_B^{(k+1)})$. In particular, the alternating optimization proceeds in a two-step fashion: At iteration k, we first fix $\widehat{B}^{(k)}$ and update $\widehat{A}^{(k+1)}$ using $\widehat{R}_A^{(k)}$; we then fix $\widehat{A}^{(k+1)}$ and update $\widehat{B}^{(k+1)}$ using $\widehat{R}_B^{(k+1)}$.

In practice, we reshape the tensor observations to a feature-by-sample matrix $X \in \mathbb{R}^{(pq)\times n}$, with feature index corresponding to the pair (i, ℓ). We then explicitly compute the full Spearman rank-correlation matrix $\hat{\rho} \in \mathbb{R}^{(pq)\times(pq)}$ using GPU parallelism to accelerate the dominant all-pairs rank-correlation cost. Concretely, we launch GPU threads over feature pairs (s, u), where each thread computes the rank correlation $\hat{\rho}_{su}$ (and writes the symmetric entry $\hat{\rho}_{us}$), thereby parallelizing the $O((pq)^2)$ pairwise dependence calculations.

The calibrated plug-in matrix $\widehat{R}$ is then viewed as the element-wise sine transform of $\hat{\rho}$ according to (3.1). However, in implementation, we do not separately materialize $\widehat{R}$ as another dense $(pq) \times (pq)$ matrix. Instead, when constructing the reduced matrices R_A and R_B, we extract the required sub-blocks from $\hat{\rho}$ according to the vectorization order and apply the sine calibration to those sub-blocks on demand.

The subsequent formation of R_A and R_B is further parallelized via multithreading. Specifically, we evaluate (3.2) and (3.3) row-wise (or column-wise) and dispatch independent index blocks to parallel workers, so that each worker constructs one row of R_A (or R_B) by repeatedly extracting the corresponding $\hat{\rho}$ sub-blocks and accumulating the weighted traces with respect to the current iterate B (or A). In practice, the proposed implementation yields measurable wall-clock gains over a naive baseline, and the benefit becomes clearer as the graph size increases.

To further support the efficiency claim above, we conducted a small-scale runtime study on subsampled concept sets with $p \in \{80, 120, 160\}$ and a fixed sample size $n = 50$. We compared a naive implementation, which uses CPU-based Spearman correlation computation and serial construction of the reduced matrices, against our implementation used in practice, which combines GPU-based rank-correlation estimation with parallel reduced-matrix construction. The reported wall-clock time measures the graph-learning pipeline, including rank-based correlation estimation and alternating precision-matrix updates.

As shown in Table 1, our implementation consistently reduced the end-to-end runtime for moderate subsampled problem sizes. Averaged over three runs, the total runtime decreased from 3.67 s to 2.55 s at $p = 120$ and from 8.12 s to 6.05 s at $p = 160$, corresponding to speedups of approximately 1.44× and 1.34×, respectively. For the smallest setting $p = 80$, the average speedup was more modest (1.18×) and exhibited larger variability across runs, suggesting that fixed implementation overhead is relatively more pronounced at very small scales.

Overall, these results indicate that the proposed implementation provides practical wall-clock improvements over a naive baseline, with clearer gains emerging as the graph size increases.

Table 1. Runtime comparison between a naive implementation and our implementation on subsampled concept sets.

#Concepts	Naive Total (s)	Ours Total (s)	Speedup
80	1.73 ± 0.07	1.53 ± 0.38	1.18×
120	3.67 ± 0.06	2.55 ± 0.06	1.44×
160	8.12 ± 0.43	6.05 ± 0.41	1.34×

Note: Reported time is wall-clock seconds (mean ± standard deviation (std) over 3 runs). Speedup is computed as Naive/Ours.

3.5. Graph-enhanced representation learning for downstream tasks

The estimated nonparanormal MNGM yields a concept-level row precision matrix $A = U^{-1} \in \mathbb{R}^{p\times p}$. We define the global binary concept graph $\mathcal{G}_c = (C, \mathcal{E}_c)$ by the off-diagonal support of A. Specifically, its adjacency matrix is $\widehat{\mathcal{A}} \in \{0,1\}^{p\times p}$ with $\widehat{\mathcal{A}}_{ij} = \mathbf{1}\{A_{ij} \neq 0\}$ for $i \neq j$ and $\widehat{\mathcal{A}}_{ii} = 0$. We use this graph as a structural prior in downstream prediction by performing message passing on the global graph and extracting document-level graph features via activated-node pooling.

For each document d_i (or bootstrap view $d_i^{(t)}$), we align its extracted concepts to the global vocabulary $C = \{c_1, \ldots, c_p\}$ and obtain an activated concept index set $S_i \subseteq \{1, \ldots, p\}$, where $j \in S_i$ indicates that concept c_j appears in the document.

Node features are given by the q-dimensional concept embeddings. Let $X \in \mathbb{R}^{p\times q}$ denote the global node feature matrix stacking embeddings for all concepts in C. We apply an L-layer GNN on the *global* graph $\mathcal{G}_c$ to obtain graph-refined concept representations $H^{(L)} \in \mathbb{R}^{p\times d}$, where d denotes hidden size:

$$H^{(\ell+1)} \;=\; \sigma\Big(\widetilde{D}^{-1/2}\,\widetilde{\mathcal{A}}\,\widetilde{D}^{-1/2}\,H^{(\ell)}W^{(\ell)}\Big), \qquad \ell = 0, \ldots, L-1, \tag{3.4}$$

where $H^{(0)} = X$, $\widetilde{\mathcal{A}} = \widehat{\mathcal{A}} + I$ adds self-loops, $\widetilde{D}$ is the corresponding degree matrix, and $\{W^{(\ell)}\}$ are learnable weights. Since $\widehat{\mathcal{A}}$ is binary, we use an unweighted topology, and edge influence is determined by normalized adjacency. This global propagation allows each concept representation to incorporate information from its conditional-dependence neighborhood in $\mathcal{G}_c$.

Let $H^{(L)} = \Phi_{\theta_{\text{gnn}}}(\widehat{\mathcal{A}}, X)$ denote the L-layer message passing output on the global graph. Rather than running message passing on a document-induced subgraph, we compute a document-level graph feature by pooling the refined representations of the activated concepts after L layers:

$$z_i^{\text{graph}} \;=\; \text{Pool}(\{\, H_j^{(L)} : j \in S_i \,\}), \tag{3.5}$$

where Pool$(\cdot)$ is a permutation-invariant operator (mean pooling in our implementation). Intuitively, $H^{(L)}$ captures global relational context from $\mathcal{G}_c$, while S_i selects the subset of concepts mentioned in the document, yielding a document-specific graph summary. The empirical distribution of the document-specific concept set size $|S_i|$ is reported in Appendix A.3, which provides additional evidence that the mean-pooling step is stable in practice.

Let z_i^{text} be the document representation from the text branch. More generally, if the raw text encoder output has dimension d_{enc}, we optionally map it through a learnable adapter to a downstream text-feature dimension d_{text}, so that

$$z_i^{\text{text}} \in \mathbb{R}^{d_{\text{text}}}.$$

Similarly, after L layers of message passing, we have

$$H^{(L)} \in \mathbb{R}^{p \times d_{\text{graph}}},$$

and the pooled graph feature satisfies

$$z_i^{\text{graph}} \in \mathbb{R}^{d_{\text{graph}}}.$$

We then fuse the two views by concatenation:

$$z_i = \text{Concat}(z_i^{\text{text}}, z_i^{\text{graph}}) \in \mathbb{R}^{d_{\text{text}}+d_{\text{graph}}}.$$

In our implementation, the frozen Qwen2 encoder first produces a 3584-dimensional sentence embedding, which is mapped by a linear adapter to $d_{\text{text}} = 256$; the GNN hidden/output dimension is $d_{\text{graph}} = 128$. Hence,

$$z_i^{\text{text}} \in \mathbb{R}^{256}, \qquad z_i^{\text{graph}} \in \mathbb{R}^{128}, \qquad z_i \in \mathbb{R}^{384}.$$

We fuse the two views by concatenation

$$z_i = \text{Concat}(z_i^{\text{text}}, z_i^{\text{graph}})$$

and predict the target as $\hat{y}_i = g(z_i)$, where $g(\cdot)$ denotes the neural network used in downstream tasks, such as an multilayer perceptron (MLP).

Downstream training keeps the estimated graph $\widehat{\mathcal{A}}$ fixed and learns the parameters of the GNN and the prediction head (optionally keeping the text encoder fixed). The final prediction is

$$\hat{y}_i = g_{\theta_g}\Big(\text{Concat}(z_i^{\text{text}}, z_i^{\text{graph}})\Big).$$

Under the loss function $\ell(\cdot)$, we solve

$$\min_{\theta_{\text{gnn}}, \theta_g} \frac{1}{N} \sum_{i=1}^{N} \ell\Big(g_{\theta_g}\Big(\text{Concat}(z_i^{\text{text}}, \text{Pool}(\{H_j^{(L)} : j \in S_i\}))\Big), y_i\Big),$$

where N denotes the number of labeled documents used in the downstream prediction task (e.g., job postings in a given training split), and $\Phi_{\theta_{\text{gnn}}}$ is defined in (3.4). This separation preserves the statistical interpretation of $\widehat{\mathcal{A}}$ as a recovered conditional-dependence structure while enabling its reuse as a structural prior in downstream prediction.

Algorithm 1 summarizes the proposed framework, highlighting the interaction between statistical graph estimation and neural representation learning.

Algorithm 1: Graph-structured representation learning algorithm

Input: Corpus $\mathcal{D} = \{d_i\}_{i=1}^{N_{doc}}$; bootstrap times T; global concept vocabulary C(size p); penalty parameters λ, ρ (or edge budget); GNN layers L.

Output: Concept graph $\mathcal{G}_c$; trained downstream predictor $g(\cdot)$.

Bootstrap and concept embedding:;

for $t = 1$ **to** T **do**

  for $i = 1$ **to** N_{doc} **do**

    Generate bootstrap document $\tilde{d}_i^{(t)}$ by shuffling sentences in paragraphs from d_i;

  Use an LLM to extract concepts C and obtain a matrix-valued embedding $Y^{(t)} \in \mathbb{R}^{p\times q}$;

Concept graph estimation:;

Estimate rank-based correlation $\widehat{R}$ from $\{\mathrm{vec}(Y^{(t)})\}_{t=1}^{T}$;

Estimate row and column precision matrices (A, B) via penalized nonparanormal likelihood, optimization via alternating updates using reduced matrices (R_A, R_B) (Section 3.4);

Construct concept graph $\mathcal{G}_c$ from the support of A.

(Here, B captures dependence across embedding dimensions and is treated as a nuisance component for graph construction.)

Graph-enhanced prediction:;

Apply a GNN on $\mathcal{G}_c$ to obtain refined concept representations $H^{(L)}$;

For each document, obtain the activated concept set S_i and pool $H_j^{(L)} : j \in S_i$ to form z_i^{graph};

Concatenate graph-based and text-based document features and train a task-specific predictor $g(\cdot)$;

return $\mathcal{G}_c$ and $g(\cdot)$;

4. Real data analysis

4.1. Experimental setup and evaluation metrics

We use salary prediction as a controlled proxy task to probe whether the recovered concept graph provides measurable statistical benefits (risk and stability), rather than treating prediction as the end goal.

The data used in this study consist of a dual-source text corpus. Specifically, we collect 45 core textbooks from statistics, data science, computer science, biostatistics, and economics and finance to represent structured academic knowledge. In addition, we gather 10,114 data-science-related job posting texts released in 2025 from major Chinese recruitment platforms, including Boss Zhipin, Zhaopin, and Liepin, covering five job categories: data analysis, software and algorithm, biomedical, economics and finance, and large language model algorithm positions (details shown in Table 2). Together, these two sources provide a unified text corpus that reflects both educational knowledge structures and labor market skill demands. In the current experiments, the concept-vocabulary construction and graph-construction stages are performed at the corpus level using unlabeled texts from the full corpus. Therefore, the downstream evaluation should be interpreted as transductive with respect to text-structure construction, although no test labels are used in any stage of predictor

training or model selection. This dataset has been uploaded to GitHub repository (https://github.com/Rayair019/Job-posting-data).

Table 2. Summary of the text corpus and concept sets.

Data source	Domain / Category	#Documents	Text length
Textbooks	Statistics, Data Science, Computer Science, Biostatistics, Economics and Finance	45	~9.6M words
Job postings	Data Analysis	2650	885,821
	Software & Algorithm	1898	636,585
	Biomedical	1939	632,898
	Economics & Finance	2095	646,820
	LLM Algorithm	1532	538,107

Prediction performance is evaluated by mean squared error (MSE) and mean absolute error (MAE) on the held-out test split of each fold, defined as

$$\mathrm{MSE} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} (y_i - \hat{y}_i)^2, \qquad \mathrm{MAE} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} |y_i - \hat{y}_i|,$$

where y_i and $\hat{y}_i$ denote the ground-truth and predicted targets, respectively, and n_{test} is the number of test samples in the fold. To assess stability across data splits, we additionally report the cross-validation variability of MSE, summarized by Std(MSE) (defined in Section 4.2). All metrics are computed per fold and then aggregated across the five folds. For downstream prediction, we include city metadata by concatenating a city embedding as an auxiliary feature; this does not affect the downstream prediction stage.

4.2. Cross-validation stability of downstream risk

Beyond average predictive accuracy, we quantify estimation stability across data splits using the variability of test risk under K-fold cross-validation. We use cross-validation variability of predictive risk as an empirical stability diagnostic, which is directly relevant to downstream reliability.

Concretely, under K-fold cross-validation, let $\mathcal{R}^{(f)}$ denote the test risk (e.g., MSE or MAE) on fold f, $f = 1, 2, \ldots, K$. We summarize stability by the standard deviation across folds:

$$\mathrm{Std}(\mathcal{R}) = \sqrt{\frac{1}{K-1} \sum_{f=1}^{K} (\mathcal{R}^{(f)} - \overline{\mathcal{R}})^2}, \qquad \overline{\mathcal{R}} = \frac{1}{K} \sum_{f=1}^{K} \mathcal{R}^{(f)}.$$

A smaller $\mathrm{Std}(\mathcal{R})$ indicates that performance is less sensitive to the particular train–test split, suggesting improved estimation stability. In the experiments reported, we consistently observe that the graph-enhanced models achieve lower fold-to-fold variability than text-only baselines, in addition to lower average MAE/MSE. This supports the view that the learned concept conditional-dependence structure acts as a stabilizing structural prior rather than a brittle source of extra model capacity.

Table 3 reports Std(MSE) together with mean MSE/MAE for each model.

Table 3. Prediction performance across text encoders and graph-learning baselines.

Model	MSE	MAE	Std(MSE)
GloVe (Text-only)	1.3861	0.8038	0.1084
LLaMA2 (Text-only)	0.9912	0.6814	0.0909
DeepSeek (Text-only)	0.9545	0.6548	0.0797
Qwen2 (Text-only)	0.8545	0.6508	0.0562
Matrix-GloVe (MNGM Graph)	1.3159	0.7797	0.1271
Qwen2 + ProGNN Graph	0.8354	0.6338	0.0480
Qwen2 + IDGL Graph	0.8370	0.6332	0.0462
npGraph-LM (Ours)	**0.8197**	**0.6290**	**0.0396**

Note: MSE and MAE are reported on the independent test set, while Std(MSE) denotes the cross-fold variability from 5-fold cross-validation on the training set.

4.3. Prediction performance across text encoders

We compare npGraph-LM with both text-only baselines under different pretrained text encoders (GloVe, Qwen2, LLaMA2, and DeepSeek) and graph-enhanced baselines using the same Qwen2 text encoder. Besides Matrix-GloVe, which estimates a concept conditional-dependence graph under a Gaussian MNGM assumption and incorporates it into the downstream predictor, we additionally consider two representative graph-learning baselines: Qwen2 + ProGNN graph and Qwen2 + IDGL graph. ProGNN refines graph structure through task-coupled graph denoising and regularization, while iterative dual graph learning (IDGL) learns task-adaptive graph connectivity directly from node features. By comparison, npGraph-LM estimates the graph under a nonparanormal MNGM via rank-based plug-in correlations and leverages LLM-derived concept embeddings. All methods share the same downstream prediction head and training protocol.

Table 3 summarizes five-fold cross-validation results. Among all compared methods, npGraph-LM achieves the lowest average MSE and MAE, outperforming not only all text-only baselines but also the two graph-learning baselines based on ProGNN and IDGL. Compared with Qwen2 (text only), the additional gains indicate that explicitly modeling concept-level conditional dependence provides complementary predictive value beyond stronger text representations alone. Compared with Matrix-GloVe, the improvement is consistent with the intended robustness benefit of relaxing the Gaussian assumption in graph estimation and replacing static GloVe-based concept representations with LLM-derived embeddings. Notably, npGraph-LM also yields the smallest cross-fold variability, with a lower Std(MSE) than both Qwen2 + ProGNN graph and Qwen2 + IDGL graph, suggesting that the performance gain is not obtained at the cost of higher split sensitivity and that the learned graph is comparatively more stable across folds. Additional sensitivity analysis with respect to the sparsity-controlling graph parameters is provided in Appendix A.2.

4.4. Category-wise analysis of structural effects

To understand where concept structure helps most and to verify that the observed gains are attributable to the *learned dependency pattern* rather than the mere presence of a graph encoder, we

report category-wise prediction errors under a controlled comparison between the learned concept graph and a random-graph baseline with matched sparsity.

For the main model, let $\widehat{\mathcal{A}} \in \{0,1\}^{p\times p}$ denote the learned binary concept adjacency, where an edge indicates conditional dependence between two concepts. We use $\widehat{\mathcal{A}}$ for message passing in the GNN encoder, and combine the resulting graph representation with the text representation for final prediction.

We further consider a random-graph control with matched sparsity. Specifically, we construct a random binary graph $\mathcal{A}_{\text{rand}} \in \{0,1\}^{p\times p}$ by uniformly sampling the same number of off-diagonal edges as in $\widehat{\mathcal{A}}$ and symmetrizing, so that $|Edge(\mathcal{A}_{\text{rand}})| = |Edge(\widehat{\mathcal{A}})|$. This preserves overall sparsity while destroying the learned conditional-dependence pattern. All other components (text encoder, concept vocabulary and node features, GNN architecture, and training protocol) are kept identical; only the adjacency used for message passing is changed.

To contextualize these structural effects, we also report a *text-only* baseline that removes message passing entirely. Table 4 shows that introducing a graph encoder is generally helpful: Both the random graph and the learned graph reduce MSE relative to text-only across nearly all job categories. However, the learned graph achieves the best overall performance relative to the matched-sparsity random control, indicating that improvements are not solely due to adding a sparse adjacency for aggregation. At the category level, the learned graph yields consistent gains in data analysis, software & algorithm, biomedical, and LLM algorithm, while economics & finance is the only case where the text-only baseline is slightly better. This pattern suggests that meaningful conditional-dependence structure contributes to performance beyond generic smoothing, although the margin can vary by domain.

Table 4. Category-wise performance (MSE).

Job category	Text-only	Random graph	Learned graph (ours)
Data Analysis	0.8949	0.8875	**0.8864**
Software & Algorithm	0.5275	0.5279	**0.5044**
Biomedical	0.3449	0.3560	**0.3392**
Economics & Finance	**1.1479**	1.1931	1.1790
LLM Algorithm	1.3059	1.1892	**1.1659**
Total	0.8422	0.8341	**0.8197**

Note: The random-graph control matches edge density and differs only in the adjacency used for message passing.

4.5. Qualitative analysis of learned concept structures

To provide further qualitative intuition for the learned concept conditional-dependence structure, we visualize a relatively sparse local subgraph extracted from the learned concept graph, as shown in Figure 3. Unlike a densely connected clique, this subgraph shows a modular but connected organization. In particular, discrete outcome serves as a bridge between a biostatistics-related group (biostatistics, biological data, bio data variability), a Bayesian modeling group (Bayesian, Bayesian network, Bayesian school), and a methodological group (orthogonal design, regularization, orthogonal polynomial regression).

This pattern suggests that the learned graph captures selective conditional-dependence relations among concept groups, rather than simply forming a dense co-occurrence network. The resulting local structure remains interpretable while being visibly sparser than a fully connected neighborhood, which is consistent with the sparsity induced by the graphical model.

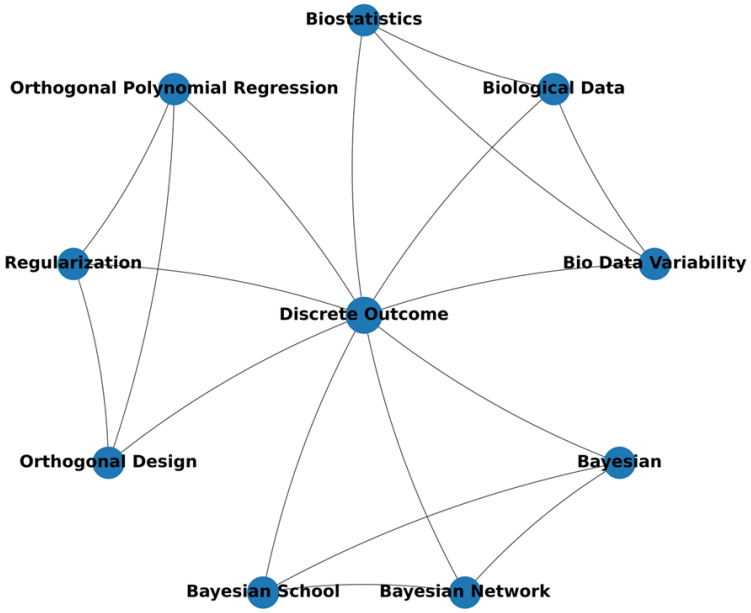
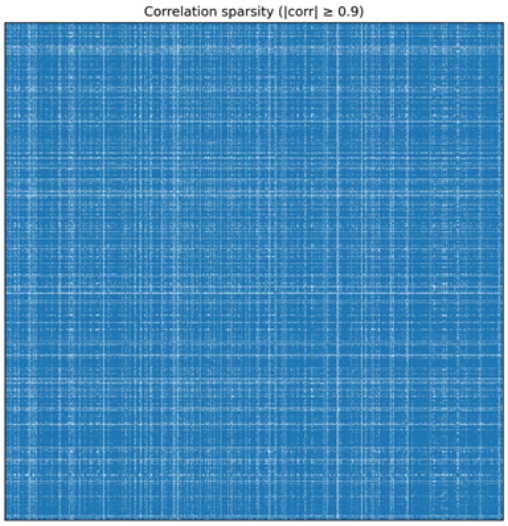


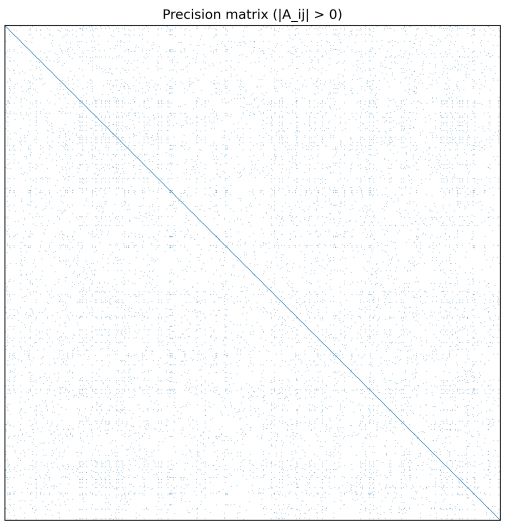
Figure 3. A relatively sparse local subgraph from the learned concept graph. The displayed subgraph contains ten connected concepts and exhibits a modular but non-fully-connected structure. The discrete outcome acts as a bridge linking a biostatistics-related group (biostatistics, biological data, bio data variability), a Bayesian modeling group (Bayesian, Bayesian Network, Bayesian School), and a methodological group (Orthogonal Design, Regularization, Orthogonal Polynomial Regression), illustrating an interpretable local conditional-dependence pattern.

To complement the subgraph visualization, we further compare the global connectivity implied by marginal correlations and by conditional dependencies under our model. Specifically, we compute the concept-concept Pearson correlation matrix from the learned concept embeddings and threshold it at a high level ($|\text{corr}| \geq 0.9$) to obtain a binary thresholded correlation pattern. As shown in Figure 4(a), the resulting pattern remains highly dense, indicating widespread marginal associations across concepts and offering limited interpretability as a sparse dependency structure. In contrast, we inspect the estimated row precision matrix $A = U^{-1}$, whose off-diagonal support encodes conditional dependencies among concepts in the nonparanormal matrix graphical model. Figure 4(b) visualizes the binary support of the entries of A, i.e., the nonzero pattern returned by the penalized estimator under the regularization parameter λ selected by cross-validation, revealing a markedly sparser pattern than the correlation baseline. This stark difference highlights that many marginal correlations are mediated by other concepts and disappear once we condition on the full concept set, whereas the precision-based graph retains only a small subset of concept pairs with direct conditional dependence. Consequently, the learned precision support provides a compact and more interpretable global

summary of the concept graph.



(a) Marginal correlation (thresholded).



(b) Precision support (nonzeros).

Figure 4. Global sparsity patterns from marginal correlation and conditional dependence. (a) Binary support of the concept–concept Pearson correlation matrix computed from the row embeddings of Y. For visualization, an entry is set to 1 if $|\mathrm{corr}(c_i, c_j)| \geq 0.9$ and to 0 otherwise, so "thresholded correlation" refers to this binary thresholding step. (b) Binary support of the estimated row precision matrix $A = U^{-1}$ from the nonparanormal matrix graphical model. An entry is set to 1 when the corresponding estimated precision coefficient satisfies the thresholding rule used in the visualization (in our main figure, nonzero under the selected regularization level λ), and to 0 otherwise. Thus, the displayed pattern represents the support induced by the penalized estimator under the cross-validated choice of λ.

4.6. Discussion

We use salary prediction as a proxy task to assess whether a concept-level conditional-dependence graph learned from LM embeddings yields measurable benefits beyond text-only representations. Empirically, the graph-enhanced model attains lower average risk (MSE/MAE) and smaller fold-to-fold variability (Std(MSE)) under five-fold cross-validation, suggesting improved downstream robustness.

The category-wise results show that the graph-enhanced model yields consistent improvements across job categories. While the magnitude of gains varies slightly by domain, the overall trend supports the benefit of incorporating the learned concept structure in downstream prediction.

The matched-sparsity random-graph control helps isolate the role of the learned dependency pattern. Keeping the encoder, node features, GNN architecture, and training protocol fixed, replacing the learned adjacency with a random adjacency increases prediction error. This suggests that the observed gains are driven by the recovered conditional-dependence structure, rather than simply by adding a graph encoder.

Qualitative visualizations provide complementary context. The local subgraph suggests sparse and interpretable connections among related concepts, and the precision-matrix sparsity pattern reflects

conditional-independence structure induced by the graphical model. These analyses are descriptive and serve primarily to illustrate typical structures learned by the method.

Overall, the results provide evidence that learning and utilizing a sparse concept conditional-dependence graph under a nonparanormal matrix graphical model can improve both predictive risk and cross-validation stability relative to text-only baselines and matched-sparsity random-graph controls.

5. Conclusions

This paper addresses the problem of learning an interpretable and reusable concept-level conditional-dependence graph from modern language representations. While matrix-normal graphical modeling provides a principled route to recover sparse row/column precision structure, modern LM-derived embeddings often deviate from Gaussianity, which can undermine both estimation robustness and downstream reuse. To bridge this gap, we proposed npGraph-LM, a nonparanormal matrix-normal graphical modeling framework for matrix-valued concept embeddings that preserves the conditional-independence interpretation of the row precision matrix while allowing unknown monotone marginal distortions.

Methodologically, npGraph-LM replaces covariance-based likelihood terms with rank-based plug-in correlations, avoiding explicit estimation of the marginal transformations and improving robustness to skewness and heavy tails. To make this estimation scalable in the matrix-normal setting, we further showed that the optimization depends on the high-dimensional plug-in matrix $\widehat{R}$ only through two reduced matrices (R_A, R_B), enabling efficient alternating updates and practical parallel implementations without materializing the full $pq \times pq$ object. The resulting concept graph, defined by the off-diagonal support of the estimated row precision matrix, provides a statistically grounded structural prior that can be integrated into downstream prediction via global message passing and activated-node pooling.

Empirically, we demonstrated that the learned concept graph can improve predictive risk and cross-validation stability on a salary-prediction task, and that the gains are attributable to meaningful conditional-dependence structure rather than a sparsity-matched random-graph control. Qualitative inspections further provided complementary evidence that the learned structure is sparse and interpretable.

A related modeling consideration is that the learned concept graph may contain heterophilic edges, since conditional-dependence relations can connect semantically different but statistically complementary concepts rather than only similar concepts. Prior work has shown that standard message passing can be less effective on heterophilic graphs, especially in node-classification settings [24, 25]. Therefore, our current use of standard message passing should be viewed as a baseline integration of the learned graph prior, rather than a claim that it is optimal for all possible concept-graph topologies. Exploring adaptive or heterophily-aware GNN architectures may further improve downstream encoding and is an important direction for future work.

Overall, npGraph-LM offers a principled and robust route to concept-structure learning from LM embeddings and enables the learned graph to be reused as an interpretable prior for downstream models. In essence, npGraph-LM relies on the existence of such a bijection to justify the monotonic transformation. How to handle cases in which this assumption fails remains an open problem and deserves further study.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by the Beijing Natural Science Foundation (L242104) and the MOE Project of Key Research Institute of Humanities and Social Sciences (22JJD110001).

Conflict of interest

The authors declare there are no conflicts of interest.

References

1. L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, et al., Training language models to follow instructions with human feedback, in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, (2022), 27730–27744.
2. J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, et al., GPT-4 technical report, preprint, arXiv:2303.08774.
3. A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, et al., Qwen2.5 technical report, preprint, arXiv:2412.15115.
4. H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, et al., Llama 2: Open foundation and fine-tuned chat models, preprint, arXiv:2307.09288.
5. R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, et al., On the opportunities and risks of foundation models, preprint, arXiv:2108.07258.
6. S. Minaee, T. Mikolov, N. Nikzad, M. Chenaghlu, R. Socher, X. Amatriain, et al., Large language models: A survey, preprint, arXiv:2402.06196.
7. N. Godey, É. de la Clergerie, B. Sagot, Is anisotropy inherent to transformers? preprint, arXiv:2306.07656.
8. J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, et al., Chain-of-thought prompting elicits reasoning in large language models, in *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, (2022), 1–14.
9. F. Stollenwerk, T. Stollenwerk, Better embeddings with coupled Adam, in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, **1** (2025), 27219–27236. https://doi.org/10.18653/v1/2025.acl-long.1321
10. H. Liu, J. Lafferty, L. Wasserman, The nonparanormal: Semiparametric estimation of high dimensional undirected graphs, *J. Mach. Learn. Res.*, **10** (2009), 2295–2328.
11. G. Di Luzio, G. Morelli, Dynamic conditional SKEPTIC, preprint, arXiv:2512.11648.
12. J. Lai, J. Yin, Learning conditional dependence graph for concepts via matrix normal graphical model, *Stat. Interface*, **17** (2024), 187–198. https://doi.org/10.4310/23-SII784

13. Y. Ning, H. Liu, High-dimensional semiparametric bigraphical models, *Biometrika*, **100** (2013), 655–670. https://doi.org/10.1093/biomet/ast009

14. L. Li, Y. Yu, W. Liang, F. Zou, A novel approach for estimating multi-attribute Gaussian copula graphical models, *Stat. Probab. Lett.*, **222** (2025), 110413. https://doi.org/10.1016/j.spl.2025.110413

15. T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, preprint, arXiv:1609.02907.

16. W. L. Hamilton, R. Ying, J. Leskovec, Inductive representation learning on large graphs, in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, (2017), 1025–1035.

17. P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in *6th International Conference on Learning Representations Iclr 2018 Conference Track Proceedings*, (2018), 1–12. https://doi.org/10.17863/CAM.48429

18. L. Chen, X. Zhu, G. Zhu, Exploiting attributes and keywords for session-based recommendation with multi-view graph neural network, *Expert Syst. Appl.*, **296** (2026), 128990. https://doi.org/10.1016/j.eswa.2025.128990

19. W. Liang, J. Cao, L. Chen, Y. Wang, J. Wu, A. Beheshti, et al., Crime prediction with missing data via spatiotemporal regularized tensor decomposition, *IEEE Trans. Big Data*, **9** (2023), 1392–1407. https://doi.org/10.1109/TBDATA.2023.3283098

20. J. Yin, H. Li, Model selection and estimation in the matrix normal graphical model, *J. Multivar. Anal.*, **107** (2012), 119–140. https://doi.org/10.1016/j.jmva.2012.01.005

21. Y. Zhang, J. Song, E. C. C. Tsang, Y. Yu, Dual-branch graph transformer for node classification, *Electron. Res. Arch.*, **33** (2025), 1093–1119. https://doi.org/10.3934/era.2025049

22. Y. Tan, S. Li, Z. Li, A privacy preserving recommendation and fraud detection method based on graph convolution, *Electron. Res. Arch.*, **31** (2023), 7559–7577. https://doi.org/10.3934/era.2023382

23. C. Yang, J. Wang, S. Wei, X. Yu, A feature fusion-based attention graph convolutional network for 3D classification and segmentation, *Electron. Res. Arch.*, **31** (2023), 7365–7384. https://doi.org/10.3934/era.2023373

24. S. Luan, C. Hua, M. Xu, Q. Lu, J. Zhu, X. Chang, et al., When do graph neural networks help with node classification? Investigating the homophily principle on node distinguishability, in *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, (2023), 1–13.

25. S. Luan, C. Hua, Q. Lu, J. Zhu, M. Zhao, S. Zhang, et al., Revisiting heterophily for graph neural networks, in *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, (2022), 1–14.

A. Appendix

A.1. Proof of Proposition 3.1

Proof (detailed sketch). Let $\widetilde{Y} := f(Y)$ denote the entry-wise transformation, i.e., $\widetilde{Y}_{ij} = f_{ij}(Y_{ij})$. By assumption,

$$\mathrm{vec}(\widetilde{Y}) \sim N_{pq}(\mathrm{vec}(M),\, V \otimes U),$$

so $\widetilde{Y} \sim MN_{p,q}(M; U, V)$, and its row precision matrix is $A = U^{-1}$.

Assume that a joint density p_{XZW} exists, and each entry-wise transform f_{ij} is strictly monotone and almost-everywhere differentiable (hence, the change-of-variables formula applies). Since f acts coordinate-wise and does not mix the coordinates across (X, Z, W), the induced map $T : (X, Z, W) \mapsto (\widetilde{X}, \widetilde{Z}, \widetilde{W})$ is a measurable bijection whose inverse is also coordinate-wise; moreover, its Jacobian matrix is block diagonal almost everywhere, so its determinant factorizes accordingly.

We justify the equivalence

$$Y^{\delta} \perp\!\!\!\perp Y^{\eta} \mid Y^{\Delta\setminus\{\delta,\eta\}} \quad \Longleftrightarrow \quad \widetilde{Y}^{\delta} \perp\!\!\!\perp \widetilde{Y}^{\eta} \mid \widetilde{Y}^{\Delta\setminus\{\delta,\eta\}}$$

using the density-based definition of conditional independence. For readability, write

$$X := Y^{\delta} \in \mathbb{R}^q, \quad Z := Y^{\eta} \in \mathbb{R}^q, \quad W := Y^{\Delta\setminus\{\delta,\eta\}} \in \mathbb{R}^{(p-2)q},$$

and similarly

$$\widetilde{X} := \widetilde{Y}^{\delta}, \quad \widetilde{Z} := \widetilde{Y}^{\eta}, \quad \widetilde{W} := \widetilde{Y}^{\Delta\setminus\{\delta,\eta\}}.$$

Since f is entry-wise strictly monotone, it induces a coordinate-wise bijection

$$(\widetilde{X}, \widetilde{Z}, \widetilde{W}) = T(X, Z, W),$$

with inverse $(X, Z, W) = T^{-1}(\widetilde{X}, \widetilde{Z}, \widetilde{W})$. Assume the joint density p_{XZW} exists (the nonparanormal model is typically stated in the absolutely continuous setting). Then, by the change-of-variables formula,

$$p_{\widetilde{X}\widetilde{Z}\widetilde{W}}(\tilde{x}, \tilde{z}, \tilde{w}) = p_{XZW}(x, z, w)\, \left|\det J_{T^{-1}}(\tilde{x}, \tilde{z}, \tilde{w})\right|, \tag{A.1}$$

where $(x, z, w) = T^{-1}(\tilde{x}, \tilde{z}, \tilde{w})$ and $J_{T^{-1}}$ is the Jacobian of T^{-1}.

Because T acts *entry-wise* (separately on coordinates of X, Z, and W), the Jacobian determinant factorizes

$$\left|\det J_{T^{-1}}(\tilde{x}, \tilde{z}, \tilde{w})\right| = J_X(\tilde{x})\, J_Z(\tilde{z})\, J_W(\tilde{w}) \tag{A.2}$$

for some nonnegative functions J_X, J_Z, J_W (products of coordinate-wise derivatives of f^{-1}).

Integrating (A.1) over $(\tilde{x}, \tilde{z})$ yields the marginal density

$$p_{\widetilde{W}}(\tilde{w}) = p_W(w)\, J_W(\tilde{w}),$$

and similarly

$$p_{\widetilde{X}\widetilde{W}}(\tilde{x}, \tilde{w}) = p_{XW}(x, w)\, J_X(\tilde{x}) J_W(\tilde{w}), \qquad p_{\widetilde{Z}\widetilde{W}}(\tilde{z}, \tilde{w}) = p_{ZW}(z, w)\, J_Z(\tilde{z}) J_W(\tilde{w}).$$

Therefore, the conditional densities satisfy

$$\begin{aligned}
p_{\widetilde{X}|\widetilde{W}}(\tilde{x} \mid \tilde{w}) &= \frac{p_{\widetilde{X}\widetilde{W}}(\tilde{x}, \tilde{w})}{p_{\widetilde{W}}(\tilde{w})} = \frac{p_{XW}(x, w)\, J_X(\tilde{x}) J_W(\tilde{w})}{p_W(w)\, J_W(\tilde{w})} = p_{X|W}(x \mid w)\, J_X(\tilde{x}), \\
p_{\widetilde{Z}|\widetilde{W}}(\tilde{z} \mid \tilde{w}) &= p_{Z|W}(z \mid w)\, J_Z(\tilde{z}), \\
p_{\widetilde{X},\widetilde{Z}|\widetilde{W}}(\tilde{x}, \tilde{z} \mid \tilde{w}) &= \frac{p_{\widetilde{X}\widetilde{Z}\widetilde{W}}(\tilde{x}, \tilde{z}, \tilde{w})}{p_{\widetilde{W}}(\tilde{w})} = \frac{p_{XZW}(x, z, w)\, J_X(\tilde{x}) J_Z(\tilde{z}) J_W(\tilde{w})}{p_W(w)\, J_W(\tilde{w})} \\
&= p_{X,Z|W}(x, z \mid w)\, J_X(\tilde{x}) J_Z(\tilde{z}).
\end{aligned}$$

Now, recall the density-based definition:

$$X \perp\!\!\!\perp Z \mid W \quad \Longleftrightarrow \quad p_{X,Z|W}(x, z \mid w) = p_{X|W}(x \mid w)\, p_{Z|W}(z \mid w) \ \text{ for a.e. } (x, z, w).$$

Using the above transformation identities, we have, for a.e. $(\tilde{x}, \tilde{z}, \tilde{w})$,

$$\begin{aligned}
p_{\widetilde{X},\widetilde{Z}|\widetilde{W}}(\tilde{x}, \tilde{z} \mid \tilde{w}) &= p_{X,Z|W}(x, z \mid w)\, J_X(\tilde{x}) J_Z(\tilde{z}) \\
&\overset{X \perp\!\!\!\perp Z|W}{=} p_{X|W}(x \mid w) p_{Z|W}(z \mid w)\, J_X(\tilde{x}) J_Z(\tilde{z}) \\
&= p_{\widetilde{X}|\widetilde{W}}(\tilde{x} \mid \tilde{w})\, p_{\widetilde{Z}|\widetilde{W}}(\tilde{z} \mid \tilde{w}),
\end{aligned}$$

which shows $\widetilde{X} \perp\!\!\!\perp \widetilde{Z} \mid \widetilde{W}$. The reverse direction follows by applying the same argument to the inverse transform T^{-1}. Hence, conditional independence among row blocks is invariant under entry-wise strictly monotone transformations.

Next, under $\widetilde{Y} \sim MN_{p,q}(M; U, V)$, we have

$$\mathrm{vec}(\widetilde{Y}) \sim N_{pq}(\mathrm{vec}(M),\ V \otimes U)$$

Hence, its precision matrix is

$$\Omega := (V \otimes U)^{-1} = V^{-1} \otimes U^{-1} = B \otimes A, \quad \text{where } A = U^{-1},\ B = V^{-1}.$$

We group the coordinates of $\mathrm{vec}(\widetilde{Y})$ by *rows* of $\widetilde{Y}$: For each $\delta \in \{1, \ldots, p\}$, define the row-block vector $\widetilde{Y}^{\delta} \in \mathbb{R}^q$ as the δ-th row of $\widetilde{Y}$ (transposed to a column vector). Then, $\mathrm{vec}(\widetilde{Y})$ can be written as a concatenation of p blocks of length q:

$$\mathrm{vec}(\widetilde{Y}) = ((\widetilde{Y}^1)^\top, \ldots, (\widetilde{Y}^p)^\top)^\top.$$

With this block partition, the (δ, η) block of the Kronecker precision $\Omega = B \otimes A$ (each block is $q \times q$) satisfies the standard Kronecker block identity:

$$\Omega_{\delta\eta} \ = \ [(B \otimes A)]_{\delta\eta} \ = \ a_{\delta\eta}\, B, \qquad \delta, \eta \in \{1, \ldots, p\}. \tag{A.3}$$

Indeed, by the definition of Kronecker product, each block is the scalar $a_{\delta\eta}$ multiplying the full matrix B.

Now, recall a basic Gaussian Markov property for block-partitioned Gaussian vectors: For a Gaussian vector $G = (G_1, \ldots, G_p)$ whose precision matrix admits the block partition $\Omega = [\Omega_{\delta\eta}]_{\delta,\eta=1}^{p}$, we have

$$G_\delta \perp\!\!\!\perp G_\eta \mid G_{\{1,\ldots,p\}\setminus\{\delta,\eta\}} \quad \Longleftrightarrow \quad \Omega_{\delta\eta} = 0 \ \text{ (the } q \times q \text{ zero matrix)}.$$

Applying this to $G_\delta = \widetilde{Y}^\delta$ yields

$$\widetilde{Y}^\delta \perp\!\!\!\perp \widetilde{Y}^\eta \mid \widetilde{Y}^{\Delta\setminus\{\delta,\eta\}} \quad \Longleftrightarrow \quad \Omega_{\delta\eta} = 0.$$

Using (A.3), $\Omega_{\delta\eta} = a_{\delta\eta}B$. Since $V \succ 0$, we have $B = V^{-1} \succ 0$, and hence B is invertible. Therefore,

$$\Omega_{\delta\eta} = 0 \quad \Longleftrightarrow \quad a_{\delta\eta}B = 0 \quad \Longleftrightarrow \quad a_{\delta\eta} = 0.$$

Consequently, for $\delta \neq \eta$,

$$\widetilde{Y}^\delta \perp\!\!\!\perp \widetilde{Y}^\eta \mid \widetilde{Y}^{\Delta\setminus\{\delta,\eta\}} \quad \Longleftrightarrow \quad a_{\delta\eta} = 0,$$

which establishes the row-graph interpretation for the matrix-normal model.

Combining the invariance in Step 1 with the Gaussian/matrix-normal property in Step 2 yields

$$Y^\delta \perp\!\!\!\perp Y^\eta \mid Y^{\Delta\setminus\{\delta,\eta\}} \quad \Longleftrightarrow \quad \widetilde{Y}^\delta \perp\!\!\!\perp \widetilde{Y}^\eta \mid \widetilde{Y}^{\Delta\setminus\{\delta,\eta\}} \quad \Longleftrightarrow \quad a_{\delta\eta} = 0,$$

which proves the claim.

A.2. Sensitivity analysis of graph sparsity parameters

To examine the sensitivity of the learned concept graph and downstream prediction performance to the sparsity-controlling parameters, we conducted a grid search over candidate graph configurations estimated under different parameter settings. These parameters directly affect the sparsity pattern of the learned precision matrices and therefore determine the number of edges and the overall density of the resulting concept graph.

For each candidate parameters pair, model selection was performed exclusively on the training set using 5-fold cross-validation. Candidate graphs were compared according to the mean validation MSE across folds. Importantly, the held-out test set was not used at any stage of parameter tuning.

In addition to predictive performance, we also recorded graph sparsity statistics, including the number of undirected edges and graph density, in order to assess the trade-off between structural parsimony and downstream utility. Table A1 summarizes the full results for all candidate graph configurations.

Overall, the results show that downstream performance varies with graph sparsity, but the best-performing configurations are concentrated in a moderate sparsity regime. Graphs that are substantially denser or substantially sparser tend to yield weaker performance, suggesting that both excessive connectivity and excessive pruning can be suboptimal. The final graph used in the main experiments was selected solely based on cross-validated training performance, and it lies in a parameter region that provides a favorable trade-off between predictive accuracy and graph sparsity. These results support the robustness of the proposed framework with respect to the sparsity-controlling parameters and justify the final parameter choice adopted in the main experiments.

Table A1. Sensitivity analysis over candidate graph configurations.

Param-1	Param-2	#Edges	Density	CV MSE↓	CV MAE↓
1.15	25	46456	0.0250	0.8674 ± 0.0539	0.6245
1.15	30	46804	0.0252	0.8670 ± 0.0537	0.6250
1.15	35	46804	0.0252	0.8690 ± 0.0506	0.6258
1.15	40	46804	0.0252	0.8682 ± 0.0520	0.6269
1.15	45	46804	0.0252	0.8793 ± 0.0531	0.6289
1.20	25	4432	0.0024	0.8682 ± 0.0536	0.6258
1.20	30	21276	0.0115	0.8664 ± 0.0501	0.6232
1.20	32	21276	0.0115	0.8709 ± 0.0524	0.6280
1.20	35	21276	0.0115	0.8815 ± 0.0534	0.6314
1.20	45	21276	0.0115	0.8623 ± 0.0517	0.6251
1.25	25	11228	0.0060	0.8683 ± 0.0513	0.6256
1.25	**30**	**5232**	**0.0028**	**0.8579 ± 0.0514**	**0.6230**
1.25	35	8076	0.0043	0.8607 ± 0.0473	0.6236
1.25	40	8076	0.0043	0.8612 ± 0.0541	0.6233
1.25	45	8076	0.0043	0.8689 ± 0.0462	0.6258
1.30	25	4116	0.0022	0.8781 ± 0.0541	0.6293
1.30	30	1036	0.0006	0.8740 ± 0.0509	0.6275
1.30	35	2916	0.0016	0.8648 ± 0.0543	0.6243
1.30	40	2916	0.0016	0.8775 ± 0.0575	0.6289
1.30	45	2916	0.0016	0.8733 ± 0.0500	0.6266

Note: For each candidate graph, we report the sparsity-related parameters, the number of undirected edges, graph density, and the mean 5-fold cross-validated downstream prediction performance on the training set. The final selected graph is highlighted in bold.

A.3. Empirical distribution of the concept set size $|S_i|$

To examine whether mean pooling over the document-specific concept set S_i could become unstable when the number of matched concepts varies widely across documents, we computed the size of S_i for every document in the dataset. Here, $|S_i|$ denotes the number of matched concepts in document i.

The resulting distribution shows that the variation in $|S_i|$ is present but not extreme. Across 10,114 documents, the mean and standard deviation of $|S_i|$ are 9.97 and 7.00, respectively, while the median is 9. The first and third quartiles are 5 and 14, indicating that the middle 50% of documents contain between 5 and 14 matched concepts. The upper tail remains moderate: The 90th, 95th, and 99th percentiles are 19, 23, and 31, respectively, and the maximum value is 58. In addition, only 236 documents (2.33%) have no matched concept. See Table A2 for details.

Overall, these results indicate that the size of S_i is not concentrated in highly sparse or highly extreme regimes for most documents. Instead, the majority of samples fall into a moderate range of concept counts. This empirical distribution suggests that the mean pooling step is reasonably stable

in practice and is unlikely to be dominated by a small number of documents with unusually small or unusually large concept sets.

Table A2. Distribution summary of the document-specific concept-set size $|S_i|$

Statistic	Value
Number of documents	10,114
Mean	9.97
Standard deviation	7.00
Median	9
Q1	5
Q3	14
90th percentile	19
95th percentile	23
99th percentile	31
Maximum	58
# documents with $\lvert S_i\rvert = 0$	236 (2.33%)

A.4. Embedding stability under paragraph bootstrap

To examine whether paragraph bootstrap substantially degrades the quality of the matrix-valued concept embeddings, we analyzed the stability of the same concept embedding across different bootstrap realizations.

Recall that each bootstrap round $t = 1, \ldots, T$ produces a matrix-valued concept embedding

$$Y^{(t)} \in \mathbb{R}^{p\times q},$$

where p is the number of concepts and q is the embedding dimension. For a fixed concept c_j, let

$$y_j^{(t)} \in \mathbb{R}^q$$

denote its embedding vector in bootstrap realization t. We measured cross-bootstrap stability by computing the cosine similarity between the same concept embedding under different bootstrap realizations:

$$\mathrm{sim}(y_j^{(t_1)}, y_j^{(t_2)}) = \frac{\langle y_j^{(t_1)}, y_j^{(t_2)}\rangle}{\|y_j^{(t_1)}\|_2 \, \|y_j^{(t_2)}\|_2}, \qquad 1 \le t_1 < t_2 \le n.$$

We then aggregated these similarities over all concepts and all bootstrap pairs.

In our data, the concept embedding tensor has shape $p \times q \times n = 1928 \times 3584 \times 50$, yielding $\binom{50}{2} = 1225$ bootstrap pairs per concept and a total of 2,361,800 same-concept cross-bootstrap similarity values. Table A3 summarizes the resulting distribution.

The results show that the same concept remains highly stable across bootstrap realizations: The average cosine similarity is 0.9895, and the median is 0.9919. Moreover, even the lower tail remains

high, with the first quartile at 0.9888 and the minimum still above 0.85. This indicates that the paragraph bootstrap introduces only mild perturbations to concept-level embeddings, rather than fundamentally altering their semantics.

Table A3. Summary of cosine similarities for the same concept embedding across different bootstrap realizations

Statistic	Value
Number of pairs	2,361,800
Mean	0.9895
Standard deviation	0.0086
Minimum	0.8585
Q1	0.9888
Median	0.9919
Q3	0.9940
90th percentile	0.9964
95th percentile	0.9971
Maximum	0.9992

AIMS Press