Statistics and Its Interface Volume 17 (2024) 187–198

Learning conditional dependence graph for concepts via matrix normal graphical model*

Jizheng Lai and Jianxin Yin†

Conditional dependence relationships for random vectors are extensively studied and broadly applied. But it is not very clear how to construct the dependence graph for unstructured data like concept words or phrases in text corpus, where the variables(concepts) are not jointly observed with i.i.d. assumption. Using the global embedding methods like GloVe, we get the 'structured' representation vectors for concepts. Then we assume that all the concept vectors jointly follow a matrix normal distribution with sparse precision matrices. With the observation of the word-word co-occurrence matrix and the GloVe construction procedure, we can test this assumption empirically. The asymptotic distribution for the test statistics is derived. Another advantage of this matrix-normal distributional assumption is that the linearly additive property in word analogy tasks is natural and straightforward.

Different from knowledge graph methods, the conditional dependence graph describes the conditional dependence structure between concepts given all other concepts, which means that the concepts(nodes) linked by edges cannot be separated by other concepts. It represents an essential semantic relationship. There is no need to enumerate all related pairs as head and tail elements of a triplet in knowledge graph regime. And the relation type in this graph is solely the conditional dependence between concepts.

A penalized matrix normal graphical model(MNGM) is then employed to learn the conditional dependence graph for both the concepts and the embedding 'dimensions'. Since the concept words are nodes in our graph with huge dimensions, we employ the MDMC optimization method to speed up the glasso algorithm. Also, the algorithm is adaptive to incremental accumulation of new concepts in text corpus. On the other hand, we propose a sentence granularity bootstrap to get 'independent' repeats of samples to enhance the penalized MNGM algorithm. We name the proposed method as Matrix-GloVe.

In simulation studies, we check that the graph learned by Matrix-GloVe is more suitable for Graph Convolutional Networks(GCN) than a correlation graph, i.e. a graph determined from the k-NN method. We employ the proposed method in two scenarios from real data. The first scenario is concept graph learning for concepts in textbook corpus. Under this scenario, two tasks are studied. One is comparing the vectors output by GloVe and other word2vec methods, i.e. CBOW and Skip-Gram, then the vectors are used by penalized MNGM. Another task is link prediction among the concepts. On both tasks, Matrix-GloVe achieves better. In the second scenario, Matrix-GloVe is applied to a downstream method i.e. GCN. For node classification tasks on the BBC and BBCSport datasets, both GCN with Matrix-GloVe and GCN with Matrix-GloVe plus Deepwalk outperform GCN with k-NN.

AMS 2000 subject classifications: Primary 62F10; secondary 62F03.
Keywords and phrases: Concept graph, Conditional dependence graph, Graph convolution network, Matrix normal graphical model, Word embedding.

1. INTRODUCTION

Conditional dependence graph, also known as conditional independence graph is a powerful tool for analyzing the relationships among random variables([6], [10], [24], [31], [34]). There, the data is usually assumed to be jointly observed and i.i.d. sampled from a population. Distributional assumptions for matrix or tensor structure can partially relieve the non-satisfaction of independent distribution assumption([17], [32]). But the conditional dependence relationship is not well defined for concept words in text corpus data, and there is no standard framework for learning this dependency structure. However, concept-level relationship mining is a heated topic in natural language processing ([18], [19], [20], [21], [26]). One line of research uses the co-occurrence and count numbers in context to infer the relationships among concepts [19], while in [21], higher order supervision of courses can help to infer the dependence among concepts. In [19], a topic model based on an admixture of Poisson Markov Random Fields(APM) is built. The dependence between words can be modeled via joint Poisson MRF parameters, from a topic angle of view. In the knowledge graph regime, the prerequisite relationship among concepts is built up by different distances like video, sentence, and Wikipedia reference distances [26]. On the other hand, the word2vec representation for words in the corpus which has semantic meaning in the embedding space

*This paper is supported by National Key Research and Development Program of China (No. 2020YFC2004900).
†Corresponding author.

([2], [13], [23]). Word2vec family of methods like Skip-Gram and CBOW [23] try to find an embedding of words into a vector space, such that the word similarity corresponds to the vector similarity. A by-product and also a rather amazing result is that the additivity property holds under certain distributional assumptions for the word2vec vectors([2], [13]).

GloVe [27] is a global embedding method from which the obtained vectors are not only meaningful in a sense of global matrix factorization but also can capture the features through a sliding window in the corpus. However, the output embedding vectors are not assumed to follow any distribution in their setting.

Given a $p \times q$ dimensional data matrix W as the output of a word2vec or GloVe output, where the p rows represent p concepts and q columns represent q dimensions in the embedding vector space. Under a joint normal distribution assumption, the matrix normal graphical model(abbreviated as MNGM, [32]) is a useful tool to mine the relationship for both rows and columns of the matrix. If one wants to apply the graphical model directly to the data matrix, it's unreasonable to assume that the different dimensions in the vector space are independent. So the matrix normal is a good remedy to incorporate the dependence structure among the dimensions in vector space into the model. We show that there are four advantages of such an assumption on the embedding vectors:

- With matrix normal distribution assumption, we can learn the conditional dependence graph for the concepts.
- The conditional means of concept vectors is linear in other given concept vectors, which means that the linearly additive property is natural in this setting.
- To reduce space and time complexity, we can record and incrementally update a sample covariance statistic to carry out the algorithm.
- Under the distribution assumption of vectors, and combining the observed word-word co-occurrence matrix, a test can be constructed to test whether this assumption holds or not.

A penalized matrix normal graphical model algorithm can then be applied to these embedding vectors. We have two main modifications here: the first one is that to handle a huge dimension of the variables(concepts), we adopt the MDMC(maximum determinant matrix completion) optimization method to speed up the underlying glasso algorithm [11]. It is proven to be able to process hundreds of thousands of variables in minutes [11]. The second one is that, to get good power for the penalized MNGM, a certain amount of repeated samples is helpful. So we develop a novel 'bootstrap' sampling method by permutating the sentences within a paragraph before a GloVe model is applied to it. Hence we repeat this procedure many times to get 'independent' samples of data matrices $W^{(b)}, b = 1, \cdots, B$. The intuition behind this kind of 'independent' sample is that sentences in a paragraph usually denote a similar meaning around a certain topic, and the exchange of sentences will not change the essential dependence among the concepts.

The Matrix-GloVe algorithm is specified for applying penalized MNGM on the embedding vectors output by GloVe embeddings. We examine this algorithm through both simulation and real data analysis. The conditional dependence graph result from Matrix-GloVe is itself of independent interest and is desirable. Also, this graph can be applied to downstream procedures. Particularly, in many application scenarios, graphs are ubiquitous in describing real-world objects and their interactions. As a powerful tool for learning on graph-structured data, Graph Neural Networks (GNNs) have been widely employed for analytical tasks across various domains. However, GNNs are highly sensitive to the quality of the given graph structures. And the provided graph is inevitably incomplete and noisy. Recent research suggests that unnoticeable, deliberate perturbation (aka., adversarial attacks) in graph structure can easily result in wrong predictions for most GNNs ([8], [33], [35]). Thus, a high-quality graph structure is often required for GNNs to learn informative representations [22]. In the simulation study, we check the network got from Matrix-GloVe performs better than correlation-based networks.

In real applications, we analyze two scenarios, one is to analyze the concepts graph for the key index words in a randomly selected 10 textbooks corpus. We first check and compare the behavior of Matrix-GloVe on the vector quality, i.e. compare whether the vectors output by GloVe is better. Secondly, we study the link prediction accuracy among concepts, using Matrix-GloVe and competing for supervised methods like GloVe+SVM. The other scenario is the node classification task on BBC and BBCSports datasets.

The paper is organized as follows. In Section 2, we briefly review the GloVe and Matrix Normal Graphical Model, make the distribution assumption on GloVe vectors, and prove some properties under the assumption which can be tested empirically. Then we propose our main algorithm Matrix-GloVe, the algorithm's incremental version, and also a sentence granularity bootstrap method. In Section 3, we apply Matrix-GloVe to the downstream algorithm Graph Convolutional Networks(GCN) and check that the graph learned by MNGM is more suitable for GCN through a simulation. In Section 4, we employ the proposed method in two scenarios from real data and test the distribution assumption empirically. Then we compare our algorithm with competing methods. In Section 5, we give a quick summary and reach the main conclusion of the paper.

2. MODELS AND METHODS

2.1 GloVe embedding

Global Vectors for Word Representation(GloVe) is a statistical model for learning word embeddings [27]. They be-

lieve that the more appropriate optimization object of embedding vectors for words is the co-occurrence probability ratio between two different words, instead of the probability of words themselves. Therefore, the GloVe model utilizing the embedding vectors can express the co-occurrence probability ratio of different target words by:

$$F\left(\left(w_i - w_j\right)^T \tilde{w}_k\right) = \frac{P_{ik}}{P_{jk}}$$

where P_{ik} means the conditional probability that word k appears in the context of words/concepts i and it can be estimated by X_{ik}/X_i. Here X_{ij} is the co-occurrence count for words/concepts i and j in a prescribed sliding window, while X_i is for word i's. w_i, w_j and $\tilde{w}_k$ are target embedding vectors.

They chose the exponential function for F, and finally get the cost function:

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T w_j + b_i + b_j - \log X_{ij})^2 \tag{1}$$

where V is the size of the vocabulary, f is a weighting function (for simplicity, we choose $f \equiv 1$ in this paper), and b_i and b_j can be treated as the estimation terms for $\log X_i$ and symmetric balancing term of word j, hence holds the relation $w_i^\top w_j \approx \log X_{ij}$. See in [27] for more details.

The model is trained only on the nonzero elements in a word-word co-occurrence matrix, so it's efficient. And it can do well over a small corpus. But there's no distributional assumption and inference on the output embedding vectors from GloVe. In this paper, we make a matrix normal distribution assumption on the output embedding matrix, when aligning all the embedding vectors of concepts.

2.2 Matrix normal graphical model

Assume the data Y as a matrix-valued random variable, we say Y follows a matrix normal distribution, if Y has a density function

$$p(Y|M,U,V) = k(U,V)\exp(-\mathrm{tr}\{(Y-M)^\top U^{-1}(Y-M)V^{-1}/2\}), \tag{2}$$

where $k(U,V) = (2\pi)^{-pq/2}|U|^{-q/2}|V|^{-p/2}$ is the normalizing constant, M is the mean matrix, U is the row covariance matrix and V is column covariance matrix. This definition is equivalent to the definition via the Kronecker product, specifically,

$$Y \sim MN_{p,q}(M;U,V) \quad \text{if and only if}$$

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

We denote the corresponding precision matrices as $A = U^{-1}$, $B = V^{-1}$ for U and V, respectively. This model assumes a particular decomposable covariance matrix for $\mathrm{vec}(Y)$ that is separable in the geostatistics context [7].

The following proposition shows that there is a graphical model interpretation for the two precision matrices A and B in the matrix normal model (2). See reference in [32].

Proposition 1. *Assume that $Y \sim MN_{p,q}(M;U,V)$. If we partition the columns of Y as $Y = (Y_1, \cdots, Y_q)$, then it holds for $\gamma, \mu \in \Gamma = \{1, \cdots, q\}$ with $\gamma \neq \mu$ that*

$$Y_\gamma \perp\!\!\!\perp Y_\mu \mid Y_{\Gamma\setminus\{\gamma,\mu\}} \text{ if and only if } b_{\gamma\mu} = 0,$$

where $B = \{b_{\alpha\beta}\}_{\alpha,\beta\in\Gamma} = V^{-1}$ is the column precision matrix of the distribution; similarly, if we partition the rows of Y as $Y = (Y^1, \cdots, Y^p)^T$, then it holds for $\delta, \eta \in \Delta = \{1, \ldots, p\}$ with $\delta \neq \eta$ that

$$Y^\delta \perp\!\!\!\perp Y^\eta \mid Y^{\Delta\setminus\{\delta,\eta\}} \text{ if and only if } a_{\delta\eta} = 0$$

where $A = \{a_{\delta\eta}\}_{\delta,\eta\in\Delta} = U^{-1}$ is the row precision matrix of the distribution.

We estimate the precision matrices $A = U^{-1}, B = V^{-1}$ in model (2) by a penalized likelihood estimation. To estimate the A and B, one can minimize the following penalized negative log-likelihood function

$$\begin{aligned}\phi(A,B) \;=\; & -q\log(|A|) - p\log(|B|) \\ & + \frac{1}{n}\sum_{k=1}^{n}\mathrm{tr}\{AY_kBY_k^T\} \\ & + \sum_{i\neq j} p_{\lambda_{ij}}(a_{ij}) + \sum_{i\neq j} p_{\rho_{ij}}(b_{ij})\end{aligned} \tag{3}$$

where $p_{\lambda_{ij}}(\cdot)$ is the penalty function for the element a_{ij} of A with tuning parameter λ_{ij}, while $p_{\rho_{ij}}(\cdot)$ is the corresponding penalty function for b_{ij} with tuning parameter ρ_{ij}. Here we use lasso penalty function $|\cdot|_1$ as $p_{\lambda_{ij}}(\cdot)$ and $p_{\rho_{ij}}(\cdot)$. We tune the penalty parameters λ_{ij} and ρ_{ij} by controlling the output amount of edges on the graph at certain level. For simplicity, we always subtract the mean from Y and assume that $M = 0$.

2.3 Local test for distributional assumption

We now construct a test statistics to test the hypothesis:

H_0 : the embedding vectors output by GloVe follow matrix normal distribution $MN_{p,q}(0;U,V)$

H_1 : not H_0

Under the above null hypothesis H_0, we have a local distributional result for each embedding vector pair w_i, w_j of words/concepts i and j. We have the following result.

Proposition 2. *Suppose that the word vectors learning from GloVe algorithm follow a matrix normal distribution(hence H_0), then the joint distribution of vectors w_i and*

w_j for concept i and concept j is

$$N\left(0, \begin{pmatrix} u_{ii}V & u_{ij}V \\ u_{ij}V & u_{jj}V \end{pmatrix}\right)$$

then $w_i^\top w_j$ follows a weighted sum of $2q$ independent χ^2 distributed random variables ξ_k ($k = 1, \cdots, 2q$) each with degree of freedom 1:

$$\lambda_1 \left(\xi_1/\alpha_1 + \xi_2/\alpha_2 + \cdots + \xi_q/\alpha_q\right)$$

$$+\lambda_2 \left(\xi_{q+1}/\alpha_1 + \xi_{q+2}/\alpha_2 + \cdots + \xi_{2q}/\alpha_q\right)$$

where $\lambda_1 = 4(u_{ii}u_{jj} - u_{ij}^2)/(u_{ii} - 2u_{ij} + u_{jj})$ and $\lambda_2 = 2(u_{ij} - \sqrt{u_{ii}u_{jj}})$. And here α_k ($k = 1, \cdots, q$) are eigenvalues of the inverse of covariance matrix V^{-1} ($= B$).

Proof. From the matrix normal distribution, we have

$$\begin{pmatrix} w_i + w_j \\ w_i - w_j \end{pmatrix} \sim N(0, \Sigma)$$

where

$$\Sigma = \begin{pmatrix} (u_{ii} + 2u_{ij} + u_{jj})V & (u_{ii} - u_{jj})V \\ (u_{ii} - u_{jj})V & (u_{ii} - 2u_{ij} + u_{jj})V \end{pmatrix}.$$

Hence

$$w_i^\top w_j = \frac{1}{4}\Big((w_i + w_j)^\top (w_i + w_j)$$

$$-(w_i - w_j)^\top (w_i - w_j)\Big)$$

From the proposition 1c.3(ii) in [29], here we take, for some R in the lemma,

$$A = \begin{pmatrix} I_q & 0 \\ 0 & -I_q \end{pmatrix} = R^{-1\top}\Lambda R^{-1}$$

and

$$B = \Sigma^{-1} = R^{-1\top}R^{-1}$$

Denote $X_1 = w_i + w_j$, $X_2 = w_i - w_j$, $X = (X_1^\top, X_2^\top)^\top$, then $Y = R^{-1}X \sim N(0, I_{2q})$. Hence $w_i^\top w_j = \frac{1}{4}X^\top AX = \frac{1}{4}Y^\top \Lambda Y$, where the λ_i in the diagonal Λ is determined by the equation $det(A - \lambda B) = 0$ by the Lemma (1c.3(ii)) in [29]. It is computed as

$$\Sigma^{-1} = \begin{pmatrix} \frac{u_{ii} - 2u_{ij} + u_{jj}}{4(u_{ii}u_{jj} - u_{ij}^2)}V^{-1} & -\frac{u_{ii} - u_{jj}}{4(u_{ii}u_{jj} - u_{ij}^2)}V^{-1} \\ -\frac{u_{ii} - u_{jj}}{4(u_{ii}u_{jj} - u_{ij}^2)}V^{-1} & \frac{u_{ii} + 2u_{ij} + u_{jj}}{4(u_{ii}u_{jj} - u_{ij}^2)}V^{-1} \end{pmatrix}$$

Solve the equation $det(A - \lambda B) = 0$ we got the solutions as the weights for the independent $\chi^2(1)$'s. Here we observe that the value of $\log(X_{ij})$ for normalized X_{ij} (adjusted by X_i and X_j) can be negative, so we choose the negative root in the final second order equation. Hence the result. □

Lemma (Proposition 1c.3(ii)in [29])**.** *Let A and B be real $m \times m$ symmetric matrices of which B is positive definite. Then there exists a matrix R such that $A = R^{-1\top}\Lambda R^{-1}$ and $B = R^{-1\top}R^{-1}$, where Λ is diagonal matrix. Let $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_m$ be roots of $|A - \lambda B| = 0$. Then the ith diagonal element of Λ is λ_i.*

GloVe is designed to fit the value of $\log(X_{ij})$ from word-word co-occurrence matrix X by learned representation vectors. And for normalized X_{ij}, fit $\log(X_{ij})$ with $w_i^\top w_j$, so we can test the distribution of $\log(X_{ij})$ to confirm the distribution hypothesis of word embeddings. For each pair (i, j) of concepts, the level-α rejection region R of this test can be written as:

$$R = \{X_{ij} : \log X_{ij} \geq (1 - \alpha)^{th}\text{quantile of the}$$

$$\text{weighted sum of } \chi^2(1)\text{s as in Proposition 2. }\}$$

In Section 4.3, we use Kolmogorov-Smirnov statistics to test whether the distribution of $\log(X_{ij})$ is significantly apart from the corresponding weighted sum of χ^2 variables.

If H_0 is rejected, one can use some nonparametric method like nonparanormal method([25]), where the Spearman's or Kendall's statistics can be applied on the embedding matrix so that the matrix normal graphical model is still applicable. But limited to the space, this line of study is not further explored in this paper.

2.4 Paragraph bootstrap

GloVe uses a context window to count the co-occurrence of pair of words, that is to say, if two words are separated by too many words (more than the size of the context window), the co-occurrence of these two words won't be counted. However, in a paragraph, two closely related words can have a relatively long distance, especially when the context is about some definition. A context window may not cover both words simultaneously when sliding over them. Hence we may lose the information of two related words.

Therefore, when we randomly shuffle the sentences in the same paragraph, we may obtain different sets of embedding vectors for concepts. This process is similar to *bootstrap resampling*. Every concept has more than one observation on the same dimensions. We get more 'samples' of concepts without enlarging the size of the corpus. By doing so, we enhance the information on the relations between two words in one paragraph in the embeddings, which can be very useful for discovering the relationship between concepts. Changing the order of sentences generally does not change the meaning of the context, and it can be considered as a different flow of expression. Hence we regard different samples generated in each bootstrap process as independent samples of word embeddings, which follow matrix normal distribution.

Denote p as the number of concepts we are interested in, q as the word embedding's vector size, and n as the times

we bootstrap the paragraph, i.e. the number of concept matrices. We propose a method named Matrix-GloVe as algorithm 1. Notice that every iteration can be run in parallel to speed up the execution process.

Algorithm 1 Matrix GloVe

Input:

corpus;

Concepts $C = \{c_1, c_2, ..., c_p\}$

Output:

Precision matrices A & B.

for iteration $i = 1, 2, \cdots, n$ **do**

Use GloVe to obtain the embeddings of concepts $W^{(i)} = \{w_1^{(i)}, w_2^{(i)}, ..., w_p^{(i)}\}$ from corpus;

Form the matrix of concepts Y_i, whose jth row $Y_{i,j\cdot} = w_j^{(i)}$;

Randomly reset the orders of sentences in every paragraph;

end for

Use $\boldsymbol{Y} = \{Y_1, ..., Y_n\}$ as input data of penalized MNGM to estimate the precision matrix A & B.

2.5 Incremental updates for penalized MNGM

The learning process of GloVe is based on an iterative function of co-occurrence word pairs. So it is naturally incremental. We just collect new co-occurrence word pairs in new documents and use the iterative function to update word embeddings on the old results. Therefore, we can easily obtain the incremental version of GloVe.

According to [32], to estimate A and B is to minimize the penalized negative log-likelihood function (3), they propose a iterative procedure to minimize this function, the second step(step A) is:

$$\hat{A}^{(i+1)} = \arg\min_{A} \left\{ -\log(|A|) + tr\left(\hat{S}_A^{(i)} A\right) + \sum_{i \neq j} p_{\lambda_{ij}^*}(a_{ij}) \right\}$$

where $\hat{S}_{\mathrm{A}}^{(i)} = 1/(nq) \sum_{k=1}^{n} (Y_k - \bar{Y}) \hat{B}^{(i)} (Y_k - \bar{Y})^\top$, $\lambda_{ij}^* = \lambda_{ij}/q$, Y_k means word embeddings from kth paragraph bootstrap, $\bar{Y} = \sum_{k=1}^{n} Y_k$.

[12] showed that the problem to maximize the penalized log-likelihood

$$\log \det \mathrm{A} - \mathrm{tr}(S\mathrm{A}) - \rho \|\mathrm{A}\|_1 \tag{4}$$

is equivalent to solve the dual problem:

$$\min_{\beta} \left\{ \frac{1}{2} \left\| U_{11}^{1/2} \beta - b \right\|^2 + \rho \|\beta\|_1 \right\} \tag{5}$$

for each column of U. Where $U = \begin{pmatrix} U_{11} & u_{12} \\ u_{12}^\top & u_{22} \end{pmatrix}$ is the estimate of A^{-1}, $S = \begin{pmatrix} S_{11} & s_{12} \\ s_{12}^\top & s_{22} \end{pmatrix}$, $b = U_{11}^{-1/2} s_{12}$, if β solves (5), then $u_{11} = U_{11}\beta$.

Thus, in step A:

$$\hat{A}^{(i+1)} = \arg\min_{A} \left\{ -\log(|A|) + tr\left(\hat{S}_A^{(i)} A\right) + \sum_{i \neq j} p_{\lambda_{ij}^*}(a_{ij}) \right\} \tag{6}$$

where $\hat{S}_{\mathrm{A}}^{(i)} = 1/(nq) \sum_{k=1}^{n} (Y_k - \bar{Y}) \hat{B}^{(i)} (Y_k - \bar{Y})^\top$, $\lambda_{ij}^* = \lambda_{ij}/q$

We denote $Y_k = \begin{pmatrix} Y_{k1} \\ y_{k2} \end{pmatrix}$. y_{k2} is the new observation of the new word.

Solving (4) is equivalent to solve the dual problem (5) $\hat{S}_A^{(i)} = \begin{pmatrix} S_{11} & s_{12} \\ s_{12}^\top & s_{22} \end{pmatrix}$, $A = \begin{pmatrix} A_{11} & a_{12} \\ a_{12}^\top & a_{22} \end{pmatrix}$, $b = U_{11}^{-1/2} s_{12}$, if β solves (5), then $u_{12} = U_{11}\beta$.

Based on this idea, we can fixed the old precision matrix A_{11}, and whenever comes a new individual (a new word), we add a new column and a new row to matrix A, and use (6) to update a_{12} and a_{22}. So we obtain a incremental version of step A. Then we assumed precision matrix B would not change much, so we fix old precision matrix B, and obtain a incremental version of MNGM, this allowed us to iterate new information based on old knowledge. When all new words are added, we get a new precision matrix $A = \begin{pmatrix} A_{11} & A_{12} \\ A_{12}^\top & A_{22} \end{pmatrix}$.

Notice that [12] started graphical lasso algorithm with $U = S + \rho I$, and the diagonal of U remains unchanged in what follows. So correspondingly, we can set $u_{22} = s_{22} + \rho$, then update u_{12} in every step. a_{12} and a_{22} are recovered by: $\hat{a}_{22} = 1/\left(u_{22} - u_{12}^\top \hat{\beta}\right)$ and $\hat{a}_{12} = -\hat{\beta}\hat{a}_{22}$.

We still need to update columns and rows in A_{11}, for new nodes and edges in the conditional dependence graph may change the old edges of the old graph.

Denote p_2 as the number of new concepts from new texts. From above, we propose incremental MNGM as algorithm 2, and incremental Matrix GloVe as algorithm 3. Iteration can also be run in parallel in algorithm 3.

When the new data dimensions are too high for glasso to run fast, [11] proposed a significantly faster algorithm for learning large-scale sparse graphical models through maximum determinant matrix completion (MDMC). We can use this algorithm to replace the glasso step (4) in MNGM, we refer to it as fastMNGM below. The algorithm requires stricter sparsity conditions to ensure its equivalence with glasso, and hence may not be as accurate as glasso in application. But it greatly speeds up the process and enables the calculation of data with higher dimensions. Two small simulation experiments are shown below.

Simulation 1: High dimension with high sparsity. With true precision matrix $A_1 \in \mathbb{R}^{4000\times4000}$, $B_1 \in \mathbb{R}^{200\times200}$, sample size generated $n = 50$, with same initial matrices and same sparsity parameter (λ for step A, and ρ for step

Algorithm 2 Incremental MNGM

Input:
the old matrix U_{11} and A_{11}
New concepts embedding matrix $\boldsymbol{Y'}$
Output:
Precision Matrix A

for iteration $i = 1, 2, \cdots, p_2$ **do**
Take $Y'_{k,i,\cdot}$ as y_{k2} in equation (6)
Obtain a $p+i-1$-dimensional vector solution $\hat{\beta}$ with (5). Fill in the corresponding row and column of U using $u_{12} = U_{11}\hat{\beta}$ and $u_{22} = s_{22} + \rho$.
Use $\hat{a}_{22} = 1/\left(u_{22} - u_{12}^{\top}\hat{\beta}\right)$ and $\hat{a}_{12} = -\hat{\beta}\hat{a}_{22}$ to fill in the corresponding row and column of A.
end for
Use glasso to update whole precision matrix A until convergence.

Algorithm 3 Incremental Matrix-GloVe

Input:
new articles;
Concepts $C = \{c_1, c_2, \cdots, c_p, c'_{p+1}, \cdots, c'_{p+p_2}\}$
Output:
Precision Matrices A & B

for iteration $i = 1, 2, \cdots, n$ **do**
Use paragraph bootstrap and incremental GloVe to acquire updated embeddings of concepts $W'^{(i)} = \{w'^{(i)}_1, w'^{(i)}_2, ..., w'^{(i)}_{p+p_2}\}$
Form the matrix of concepts Y'_i, whose jth row $Y'_{i,j,\cdot} = w'^{(i)}_j$
Randomly reset the orders of sentences in every paragraph
end for
Use $\boldsymbol{Y'} = \{Y'_1, ..., Y'_n\}$ to estimate the precision matrix A' through incremental MNGM

B, same notation as [32]. Define the non-zero entry in sparse precision matrix A_1, excluding diagonal elements, as 'positive'. A_1 has 5360 positive elements.

Simulation 2: Low dimension with low sparsity. With true precision matrix $A_2 \in \mathbb{R}^{400\times400}$, $B_2 \in \mathbb{R}^{200\times200}$, sample size generated $n = 50$, with same initial matrices and same sparsity parameter (λ for step A, and ρ for step B). A_2 has 4122 positive elements.

Table 1 and table 2 shows the results executed on a standard laptop computer. When sparsity is high enough, and data dimensions are high, fastMNGM is able to achieve a comparable result with a much faster speed.

2.6 Linearly additive property

Let vec($\boldsymbol{A}$) be the vectorization of a matrix $\boldsymbol{A}$ obtained by stacking the rows of the matrix $\boldsymbol{A}$ on top of one another. Let c_1 be the target concept, whose embedding is w_1, and $c_2, c_3, ..., c_n$ be all the other concepts, whose embeddings are $w_2, w_3, ..., w_n$. We want to know if c_1's embedding can be expressed linearly by other embeddings. Namely $w_1 = \alpha_2 w_2 + \alpha_3 w_3 + ... + \alpha_n w_n + d$, where d is a constant.

Table 1. Simulation 1 Result (High dimension with high sparsity, $A_1 \in \mathbb{R}^{4000\times4000}$)

Algorithm	λ	ρ	Precision	Recall	Iteration Time
MNGM	41.97	34.27	99.55%	99.96%	37 min 51 s
fastMNGM	41.97	34.27	92.99%	86.11%	1 min 8 s
MNGM	38.97	31.82	99.51%	99.96%	38 min 11 s
fastMNGM	38.97	31.82	84.87%	93.58%	1 min 29 s

Table 2. Simulation 2 Result (Low dimension with low sparsity, $A_2 \in \mathbb{R}^{400\times400}$)

Algorithm	λ	ρ	Precisiony	Recall	Iteration Time
MNGM	7.76	7.34	97.78%	98.66%	6 s
fastMNGM	7.76	7.34	8.03%	99.61%	11 s
MNGM	10.34	9.79	100%	93.99%	7 s
fastMNGM	10.34	9.79	8.50%	95.91%	12 s

Proposition 3. *Suppose that the stacked embedding vectors of concepts follow a matrix normal distribution,* $W = \begin{pmatrix} w_1^{\top} \\ w_2^{\top} \\ ... \\ w_n^{\top} \end{pmatrix} \sim MN_{n,q}(M; U_n, V)$, $M = \begin{pmatrix} m_1^{\top} \\ m_2^{\top} \\ ... \\ m_n^{\top} \end{pmatrix}$ *and* $U_n = \begin{pmatrix} u_{11} & U_{12} \\ U_{21} & U_{22} \end{pmatrix}$.

Then for a concept c_1, whose embedding is w_1, we have

$$E(w_1^{\top} - m_1^{\top} | w_2, ..., w_n) = \beta_1(w_2^{\top} - m_2^{\top}) + ... + \beta_n(w_n^{\top} - m_n^{\top})$$

where $\beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ ... \\ \beta_n \end{pmatrix} = (U_{12} \times U_{22}^{-1})^{\top}$

Proof. $W = \begin{pmatrix} w_1^{\top} \\ W_{\setminus 1,\cdot} \end{pmatrix} \in \mathbb{R}^{n\times q}$. As we know, $\text{vec}(W^{\top}) \in \mathbb{R}^{nq\times1} \sim MN_{nq}(\text{vec}(M^{\top}), U_n \otimes V)$. Similarly, $\text{vec}(W_{\setminus 1,\cdot}^{\top}) = \begin{pmatrix} w_2 \\ ... \\ w_n \end{pmatrix} \sim MN_{(n-1)q}(\begin{pmatrix} m_2 \\ ... \\ m_n \end{pmatrix}, U_{22} \otimes V)$.

Assume that U_{22}^{-1} exists, use the conditional probability of joint normal distribution:

$$\begin{aligned} E(w_1^{\top} | w_2, ..., w_n) &= (W_{\setminus 1,\cdot} - M_{\setminus 1,\cdot})^{\top} \times (U_{12} \times U_{22}^{-1})^{\top} + m_1^{\top} \\ &= [\begin{pmatrix} w_2^{\top} \\ ... \\ w_n^{\top} \end{pmatrix} - \begin{pmatrix} m_2^{\top} \\ ... \\ m_n^{\top} \end{pmatrix}]^{\top} \times \beta + m_1^{\top} \\ &= \beta_1(w_2^{\top} - m_2^{\top}) + ... + \beta_n(w_n^{\top} - m_n^{\top}) + m_1^{\top} \end{aligned}$$

Hence the result. □

From Proposition 3 we can see, $E(w_1|w_2, w_3, ..., w_n)$ is a linear combination of embedding vectors of all other words. The coefficient vector β can be calculated by $(U_{12} \times U_{22}^{-1})^\top$. However, the U_{22}^{-1} can be hard to calculate when n is large. Here we propose a simplified algorithm. Denote J as $\begin{pmatrix} 1 & 0 \\ 0 & U_{21} \end{pmatrix}_{n\times 2}$, denote K as $\begin{pmatrix} 0 & U_{12} \\ 1 & 0 \end{pmatrix}_{2\times n}$, so $JK = \begin{pmatrix} 0 & U_{12} \\ U_{21} & 0 \end{pmatrix}_{n\times n}$, Hence $(U - JK)^{-1} = \begin{pmatrix} u_{11} & 0 \\ 0 & U_{22} \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{u_{11}} & 0 \\ 0 & U_{22}^{-1} \end{pmatrix}$. So we can obtain U_{22}^{-1} by calculating $(U - JK)^{-1}$. According to *Sherman-Morrison-Woodbury formula*:

$$\begin{aligned}(U - JK)^{-1} &= U^{-1} + U^{-1}J\left(I - KU^{-1}J\right)^{-1}KU^{-1} \\ &= A + AJ(I - KAJ)^{-1}KA.\end{aligned}$$

where A is considered known, and $(I - KAJ)$ is a 2×2 matrix, so the calculation of inversion is greatly reduced. Proposition 3 tells us that in context, one embedding vector of one concept can be linearly expressed by other concepts' vectors, which is meaningful in semantic space.

Here we pick two examples from the real data analysis task of textbook corpus for illustration:

$$\begin{aligned}&'\text{modified likelihood}' \\ =&0.027 \times '\text{likelihood}' \\ +&0.349 \times '\text{modified profile likelihood}' \\ -&0.074 \times '\text{profile likelihood}'\end{aligned}$$

$$\begin{aligned}&'\text{Jeffreys prior density}' \\ =&0.249 \times '\text{Jeffreys prior}' \\ +&0.322 \times '\text{prior density}'\end{aligned}$$

3. APPLY MATRIX-GLOVE TO GCN

With feature matrices (embedding matrices), we can obtain a precision matrix, which represents the conditional dependence graph. This graph can be very useful in many downstream tasks combined with graph neural networks.

DeepWalk is a two-stage method. In the first stage, it traverses the network with random walks to infer local structures by neighborhood relations. In the second stage, it uses Skip-Gram to learn embeddings that are enriched by the inferred structures. Therefore, we get new embeddings of nodes in the undirected graph. And with these new embeddings of nodes, we can use Matrix-GloVe again to obtain a new undirected graph. See Figure 1.

3.1 Simulation

We generate three groups of samples: G_1, G_2, G_3, assigned three labels separately. $G_1 \in \mathbb{R}^{300\times 200}, G_2 \in \mathbb{R}^{300\times 200}, G_3 \in \mathbb{R}^{300\times 200}$, $G = \begin{pmatrix} G_1 \\ G_2 \\ G_3 \end{pmatrix} \in \mathbb{R}^{900\times 200}$, and $G \sim MN_{900,200}(0, \Sigma_0 \otimes V)$, where $\Sigma_0 = \begin{pmatrix} \Sigma_1 & 0 & 0 \\ 0 & \Sigma_2 & 0 \\ 0 & 0 & \Sigma_3 \end{pmatrix}$. $\Sigma_1, \Sigma_2, \Sigma_3$ and V are randomly generated. Then we generated the observation matrix $Y \in \mathbb{R}^{900\times 200\times 20}$ from the distribution of G. The Σ_0 can be viewed as the true graph. And the labels of sample are completely depended on Σ_0.

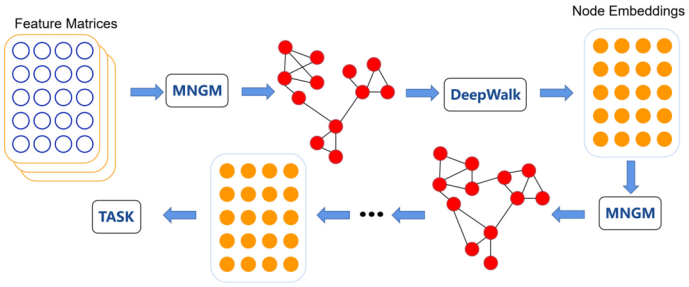

Figure 1. Flowchart of Matrix-GloVe+Deepwalk+GCN. The feature matrices can be output vectors of GloVe.

We apply MNGM+Deepwalk+GCN on the observation matrices and the node classification as a downstream task. As a comparison, we apply GCN on a k-NN graph, and also apply MNGM+GCN on the same data. Table 3 shows that the MNGM graph is better than the k-NN graph in doing such node classification, showing that the graph learned by penalized MNGM is more suitable for GCN than a k-NN graph.

Table 3. Simulation Result

Algorithms	Accuracy
k-NN+GCN	40%
MNGM+GCN	92%
MNGM+Deepwalk+GCN	**94%**

4. EMPIRICAL STUDY

4.1 Corpus

Scenario 1: concept graph learning for textbooks. We collected 10 classic books on statistics as our training corpus, including *Statistical Models by Davison*, *A First Course in Probability by Ross*, etc. The vocabulary size is about 39,000 after we remove the stop words. The input of the GloVe model should be text file, so our files are all *.txt* files. Table 4 shows the textbook list.

Books about statistics can have a lot of digits and mathematical symbols, which makes it difficult to learn their semantics. So we remove those characters that are not English letters. This helps GloVe to avoid noise from digits

Table 4. Textbook List

Index	Book Information
1	Agresti, A. (2002). *Categorical data analysis.* Wiley-Interscience. [1]
2	Blitzstein, J. K. and Hwang, J.(2014). *Introduction to probability.* CRC Press. [3]
3	Boyd, S.P. (2004). *Convex optimization.* Cambridge, U K: Cambridge University Press. [4]
4	Buşoniu L., Babuška R., De Schutter B. and E-rnst D. (2010). *Reinforcement Learning and Dynamic Programming Using Function Approximators.* Scopus. [5]
5	Givens, G. H.(2013). *Computational statistics.* John Wiley & Sons, Inc., Hoboken, NJ [14]
6	Davison, A. C.(1993). *Statistical Models.* Cambridge University Press. [9]
7	Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep learning.* MIT Press, Cambridge, MA. [15]
8	Harrell, F. E. J. *Regression modeling strategies: w-ith applications to linear models, logistic and ordinal regression, and survival analysis.* Cham: Springer. [16]
9	Racine, J.S. (2008). *Nonparametric Econometrics: A Primer.* Quantile. [28]
10	Ross, S. M. (2009). *A first course in probability, 8th edition* Pearson Prentice Hall. [30]

and mathematical symbols. For the sake of unity, the upper case letters are changed to lower case letters.

We use GloVe model to obtain the word embeddings of the concepts we are interested in, referred to as target concepts. As we know, GloVe only learns the embeddings of every single word in the corpus, but a concept may be composed of multiple words. Therefore, we combine the words of every target concept into one single word by simply concatenating them. So we can accurately get the embedding of each concept. For example, the concept '*random variable*' is replaced by '*randomvariable*', so that we can get one precise word embedding for '*randomvariable*'. These concepts are derived from the index table at the end of each book, which lists all the important indexes mentioned in the book.

Scenario 2: node classification on the BBC and BBCSport datasets. BBC Datasets are two news article datasets, originating from BBC News, provided for use as benchmarks for machine learning research. BBC dataset consists of 2225 documents from the BBC news website corresponding to stories in five topical areas from 2004-2005. It has 5 class labels: business, entertainment, politics, sport, and tech. BBCSport dataset Consists of 737 documents from the BBC Sport website corresponding to sports news articles in five topical areas from 2004-2005. It has 5 class labels: athletics, cricket, football, rugby, and tennis.

4.2 Algorithm settings

We set the GloVe context window size as 20, and the embedding size q as 200. For target concepts in scenario 1, we chose 802 concepts from the index tables of *A First Course in Probability* by *Ross* and *Statistical Models* by *Davison*.

For paragraph bootstrap, we separate paragraphs by punctuation from '.', '!' or '?' plus a newline character. And we separate sentences by a '.', '!' or '?'. Then we randomly shuffle the sentences in every paragraph for $n = 50$ times, so we get 50 text files.

To evaluate the output of the model, we use an evaluation set annotated and checked manually by some doctoral students and teachers of Renmin University. The relations between some concept pairs are annotated 0 / 1. The evaluation set includes 822 pairs of concepts, of which 306 pairs are marked as 1 and 416 pairs are marked as 0. For example, 'correlation, covariance, 1' and 'correlation, central limit theory, 0'.

In each evaluation, note TP as the number of concept pairs marked with 1 on the evaluation set predicted as 1; FN is the number of concept pairs marked as 1 on the evaluation set predicted as 0; FP is the number of concept pairs marked 0 on the evaluation set predicted to be 1, and TN is the number of concept pairs marked 0 on the evaluation set predicted to be 0.

Calculate the following three indicators for each evaluation:

$$accuracy = \frac{TP + TN}{TP + FN + FP + TN}$$

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

F1 score is an index that takes into account the accuracy and recall of the classification model.

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

We use the F1 score as the standard for adjusting parameters and as a standard to evaluate the quality of the model.

4.3 Test for matrix normal hypothesis

For embeddings learned from scenario 1, we randomly select a concept from all target concepts (happens to be 'normal distribution'), draw histograms of certain dimensions of 50 observations. According to the matrix normal hypothesis, the sample distribution should be similar to the normal distribution. See Figure 2.

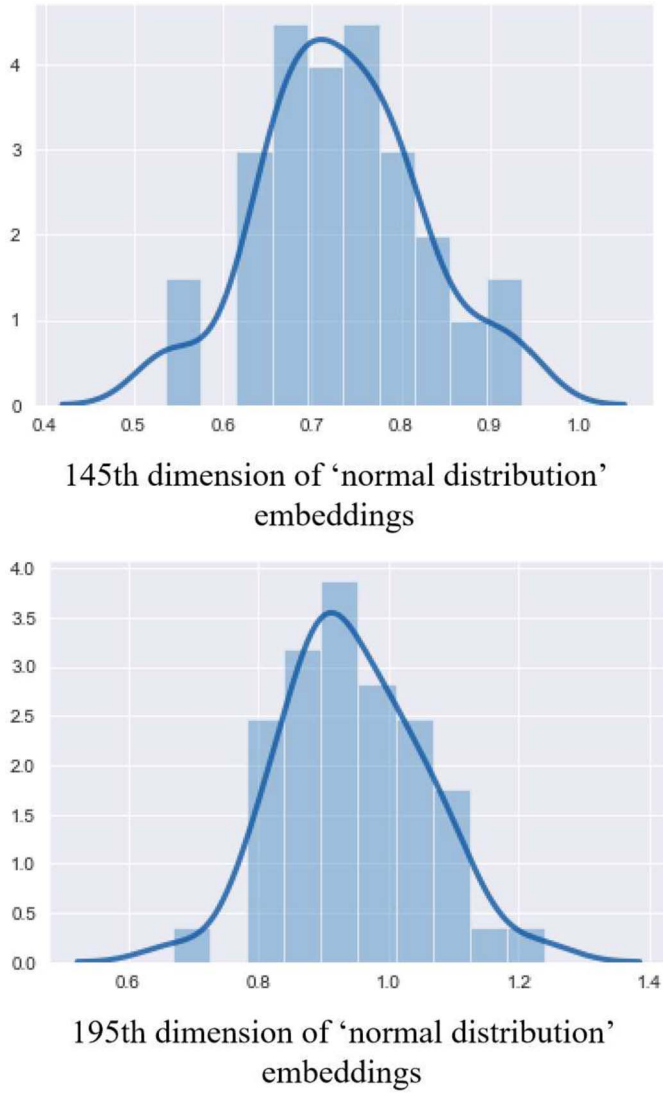

Figure 2. Histograms of 2 dimensions of embeddings of 'normal distribution'.

On the other hand, according to the matrix normal distribution, the embedding of two concepts follow:

$$\begin{pmatrix} w_i^\top \\ w_j^\top \end{pmatrix} \sim N\left(0, \begin{pmatrix} u_{ii} & u_{ij} \\ u_{ji} & u_{jj} \end{pmatrix} \otimes V\right)$$

where U and V are covariance matrices representing the dependencies of rows and columns respectively. Therefore, the covariance matrix of $(w_i, w_j)^\top$ should be:

$$\begin{pmatrix} u_{ii}V & u_{ij}V \\ u_{ji}V & u_{jj}V \end{pmatrix}$$

Therefore, the four sub matrices should be proportional. If we plot heat maps for this covariance matrix, the four blocks should show similar patterns. Therefore, if we draw the heat map of sample covariance matrices of $(w_i, w_j, w_k)^\top$ for the randomly selected i, j, k, so the 3×3 sub blocks should be proportional. Figure 3 gives us some confidence in the distribution hypothesis.

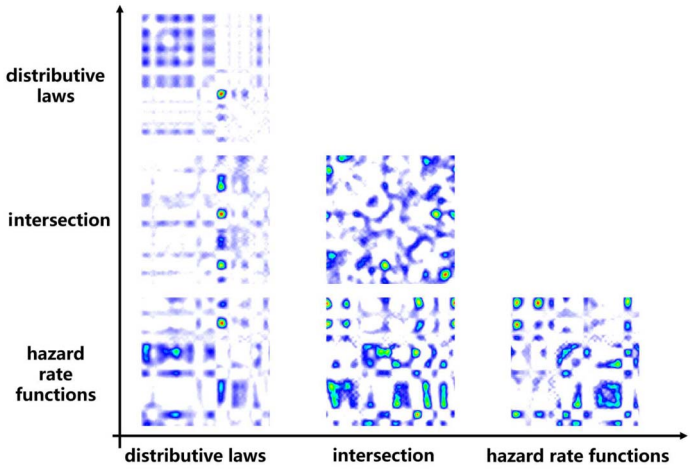

Figure 3. Heatmap of 3 random concepts' sample covariance matrix, diagonal elements are removed for better view.

We also did a test for $\log(X_{ij})$ according to Proposition 2. We randomly generate $2q$ independent random variables from χ^2 distribution with a degree of freedom of 1, and use them to simulate the theoretical values of $w_i^\top w_j$. Then we compare the simulated 'theoretical' distribution with the empirical distribution of $\log(X_{ij})$ obtained from the co-occurrence in the corpus. Since we executed paragraph bootstrap 50 times, we got 50 sample values. We plotted Q-Q plots of simulated data and 50 $\log(X_{ij})$ for a randomly chosen concept pair. If the two distributions are similar, the image should be close to a straight line. From Figure 4 we can see that the points are roughly in a straight line, consistent with the hypothesis. We also did a Kolmogorov-Smirnov test between two distributions for this pair, of which the p-value is 0.84. Therefore, the H_0 in Section 2.3 cannot be significantly rejected. Across all pairs of concepts, we did tests for 5568 pairs of (i, j) such that $\log(X_{ij}) > 0$, and 5095 of them were not rejected with a significant level $\alpha \leq 0.05$. These results give us confidence in applying the Matrix-GloVe method to this corpus.

4.4 Concept graph learning

We trained word embeddings through GloVe model, over the textbook corpus. Then we bootstrap the paragraphs 50 times, getting $\boldsymbol{M} \in \mathbb{R}^{802\times200\times50}$. We feed $\boldsymbol{M}$ to Matrix Normal Graphical Model separately, and output the estimation of precision matrix $\hat{A}$,which is sparse for a 802×802 size matrix. The sparse parameter λ_{ij} is chosen by cross validation. The matrix $\hat{A}$ has 136947 nonzero elements apart from diagonal elements. The number can also be the number of relations we find through the matrix GloVe model.

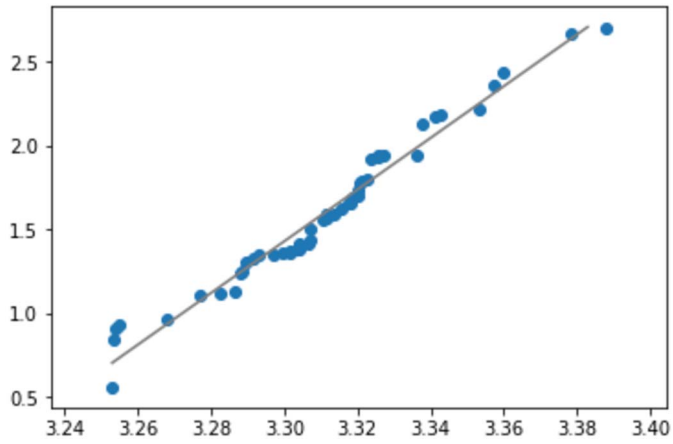

Figure 4. Q-Q plot of the simulation distribution and log(Xij) sample distribution. The horizontal axis is the quantile of log(Xij) sample, and the vertical axis is the quantile of the simulation data.

A glimpse of the concepts graph constructed from $\hat{A}$ is shown in Figure 5. Notice the edges means conditional dependence, not traditional similarity. For instance, '*Pearson's statistic*' is not connected to '*F statistic*', although they share some similarities in semantics. But both of them are connected to '*statistic*', meaning that they build relationships through the concept 'statistic'.

We compares the vectors output by GloVe, CBOW and Skip-Gram, all with MNGM as a tool to learn the structure of target concepts, and evaluated through evaluation set of annotated concepts pairs. Matrix-GloVe, CBOW MNGM and Skip-Gram MNGM are all trained on the textbook corpus, and adjusted the parameters through cross validation. Table 5 shows that Matrix-GloVe outperforms other two methods.

We also compares Matrix-GloVe with an SVM model on the GloVe vectors of given concepts pair. With GloVe embeddings of two given concepts, GloVe-SVM connects the two embeddings to predict whether the two concepts are dependent. Table 6 shows that Matrix-GloVe outperforms GloVe-SVM.

4.5 Node classification task

The node classification task is one where the algorithm has to determine the labeling of samples (represented as nodes) by looking at the labels of their neighbors. Node classification models aim to predict non-existing node properties (known as the target properties) based on other node properties. Typical models used for node classification consist of a large family of graph neural networks.

Traditional node classification requires an initial graph as the basis of graph neural networks. This can be achieved through the known network in the database. However, sometimes there does not exist a known network in the database.

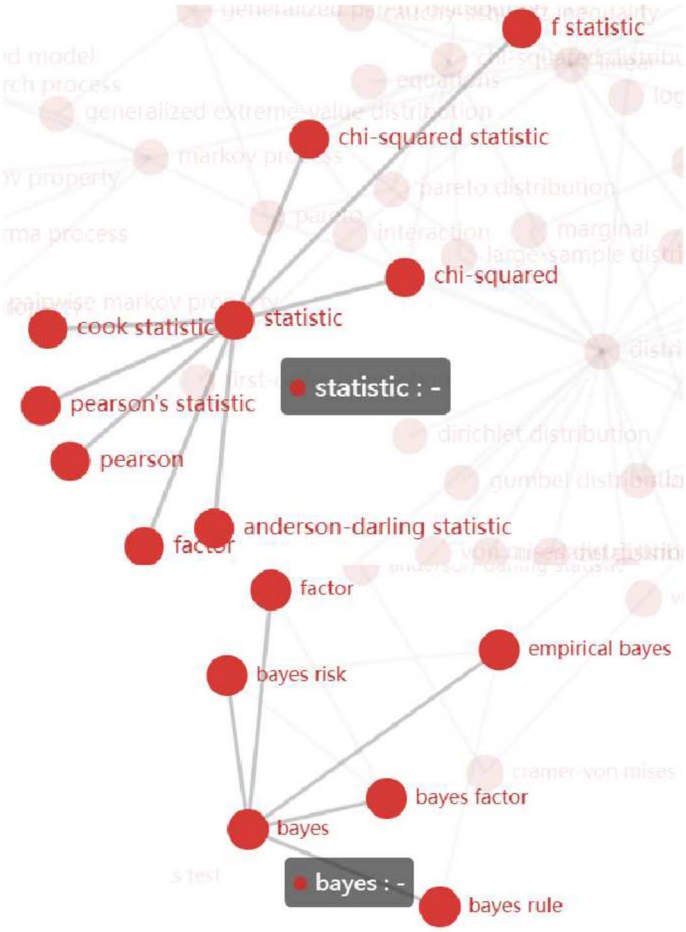

Figure 5. A glimpse of the concepts graph from the ouptput of Matrix-GloVe.

Table 5. Model performance on test set

Model	Accuracy	Precision	Recall	F_1
CBOW MNGM	56.6 %	44.4 %	**67.2** %	53.5 %
Skip-Gram MNGM	71.4 %	66.9 %	46.4 %	54.6 %
Matrix-GloVe	**72.4** %	**67.6** %	49.9 %	**57.2** %

Table 6. Matrix-Glove & SVM on test set

Model	Accuracy	Precision	Recall	F_1
GloVe-SVM	55.2%	47.6%	30.5%	37.1%
Matrix-GloVe	**72.4** %	**67.6** %	**49.9** %	**57.2**

Some would use certain easy ways, like a k-NN graph, to construct an initial graph based on node embeddings. Instead, we use MNGM Graph as the initial graph, and a simple graph neural network model to do the node classification task.

We first use GloVe and paragraph Bootstrap on BBC and BBCSport datasets to get observation matrices separately. Treating each article as a node, we apply Matrix-

Table 7. Node Classification Result

Algorithms	Acc on BBC	Acc on BBCSport
GloVe+k-NN+GCN	86.3%	80.6%
Matrix-GloVe+GCN	91.6%	82.3%
Matrix-GloVe +Deepwalk+GCN	**93.8%**	**86.5%**

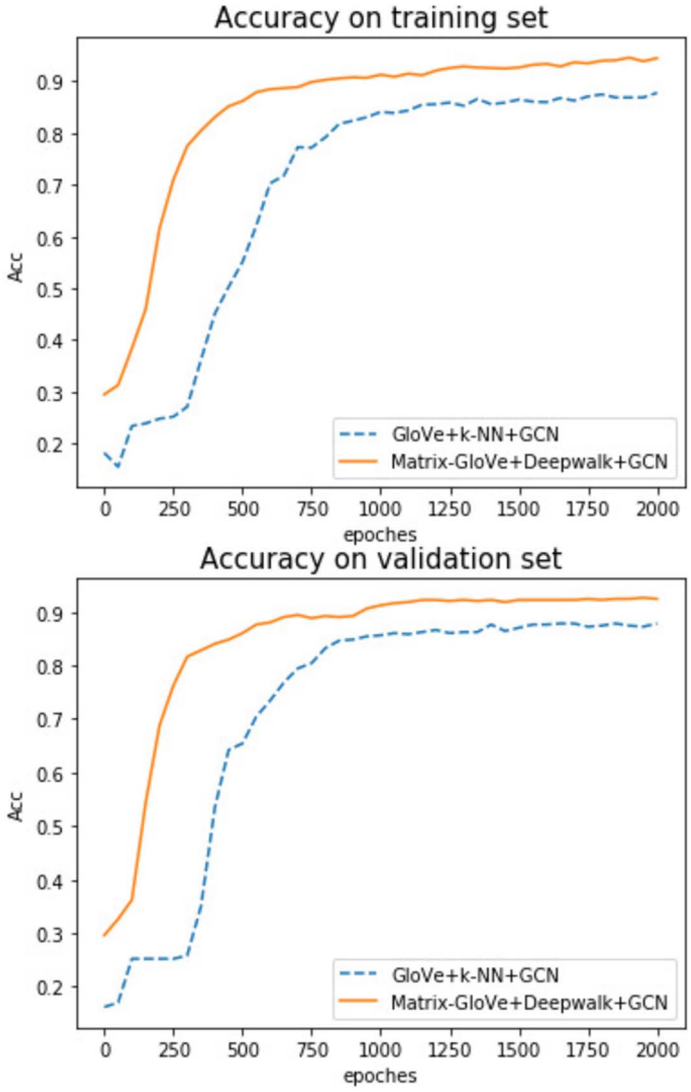

Figure 6. training process on BBC dataset.

GloVe+Deepwalk+GCN on both observation matrices for node classification tasks. As a comparison, we apply GCN on a k-NN graph of observation matrices.

The BBC dataset training set includes 1000 documents; validation set has 500 documents; 725 documents are left for test set. The BBCSport dataset training set includes 300 documents; validation set has 200 documents; 237 documents are left for test set.

From Table 7 and Figure 6 we can see that 'Matrix-GloVe+Deepwalk+GCN' achieves better performance and faster convergence speed than 'GloVe+k-NN+GCN' in node classification.

5. CONCLUSION

We developed the Matrix-GloVe, a framework to automatically learn the conditional dependence graph for unstructured data like concept words or phrases in the text corpus, where the variables (concepts) are not jointly observed with i.i.d. assumption. The graph describes the conditional dependence structure between concepts given other concepts, which means that the concepts (nodes) linked by edges cannot be separated by other concepts, representing an essential semantic relationship. Under the assumption that all the concept vectors learned from GloVe jointly follow a matrix normal distribution with sparse precision matrices, we can test this hypothesis empirically. We found the distribution for the $\log(X_{ij})$ is weighted sum of $\chi^2(1)$s, where X_{ij} is the co-occurrence counts for words i and j. We also show that the linearly additive property holds under this assumption. In Matrix-Glove, we employ the MDMC optimization method to speedup the glasso algorithm when dealing with huge dimensional data. Also, the algorithm is adaptive to incremental accumulation of text corpus. On the other hand, we developed a sentence granularity bootstrap to get 'independent' repeats of samples to help enhance power.

In concept graph learning task, both from the vector quality and the link prediction perspective, Matrix-GloVe outperforms not only related word2vec methods, like CBOW and Skip-Gram, but also the SVM+GloVe vectors. In the application of Matrix-GloVe to downstream algorithms, we found that the conditional dependence graph learned by Matrix-Glove is more suitable for GCN than other correlation graphs like k-NN. But the GloVe embedding and MNGM in Matrix-GloVe serve as two parts, how to integrate them into a whole algorithm is left as a future study.

Received 30 September 2022

REFERENCES

[1] Agresti, A. (2002). *Categorical Data Analysis.* Wiley-Interscience. MR1914507

[2] Arora, S., Li, Y., Liang, Y., Ma, T. and Risteski, A. (2016). A latent variable model approach to pmi-based word embeddings. *Transactions of the Association for Computational Linguistics* **4** 385–399.

[3] Blitzstein, J. K. and Hwang, J.(2014). *Introduction to Probability.* CRC Press. MR3237118

[4] Boyd, S.P. (2004). *Convex Optimization.* Cambridge, UK: Cambridge University Press. MR2061575

[5] Buşoniu L., Babuška R., De Schutter B. and Ernst D. (2010). *Reinforcement Learning and Dynamic Programming Using Function Approximators.* Scopus.

[6] Chung, J., Jackson, B., Mcdowell, J. and Park, C. (2022) Joint estimation and regularized aggregation of brain network in FMRI data. *Journal of Neuroscience Methods.* **364** 109374.

[7] Cressie, N. (1993). *Statistics for Spatial Data.* Wiley, New York. MR1239641

[8] Dai, H., Li, H., Tian, T., Huang, X., Wang, L., Zhu, J. and Song, L. (2018). Adversarial attack on graph structured data. *ICML*
[9] Davison, A.C. (1993). *Statistical Models.* Cambridge University Press. MR1998913
[10] Fan, J., Feng, Y. and Xia, L. (2020). A Projection-based Conditional Dependence Measure with Applications to High-dimensional Undirected Graphical Models. *Journal of Econometrics* **218(1)** 119–139. MR4111748
[11] Fattahi, S., Zhang, R. Y. and Sojoudi, S. (2019). Linear-Time Algorithm for Learning Large-Scale Sparse Graphical Models. *IEEE Access* **7** 12658–12672.
[12] Friedman, J., Hastie, T. and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics.2008 Jul.* **9(3)** 432–41.
[13] Gittens, A., Achlioptas, D. and Mahoney, M. W. (2017). Skip-Gram - Zipf + Uniform = Vector Additivity. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*
[14] Givens, G. H.(2013). *Computational statistics.* John Wiley & Sons, Inc., Hoboken, NJ MR3236433
[15] Goodfellow, I., Bengio, Y. and Courville, A.(2016). *Deep learning.* MIT Press, Cambridge, MA. MR3617773
[16] Harrell, F. E. J. *Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis* Cham: Springer.
[17] He, S., Yin, J., Li, H. and Wang, X. (2014). Graphical model selection and estimation for high dimensional tensor data. *Journal of Multivariate Analysis* **128** 165–185. MR3199836
[18] Hogan, A., Blomqvist, E., Cochez, M., D'amato, C., Melo, G.D., Gutierrez, C., Kirrane, S., Gayo, J.E.L., Navigli, R., Neumaier, S. and et al. (2021). Knowledge Graphs. *ACM Computing Surveys* **54** 1–37
[19] Inouye, D. I., Ravikumar, P. and Dhillon, I. S. (2014). Admixture of poisson MRFs: A topic model with word dependencies. *International Conference on Machine Learning.* International Machine Learning Society (IMLS).
[20] Ji, S., Pan, S., Cambria, E., Marttinen, P. and Yu, P. S. (2021). A survey on knowledge graphs: representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems.* **33(2)** 494–514 MR4378310
[21] Liu, H., Ma, W., Yang, Y. and Carbonell, J. (2016). Learning Concept Graphs from Online Educational Data. *Journal of Artificial Intelligence Research* **55** 1059–1090.
[22] Luo, D., Cheng, W., Yu, W., Zong, B., Ni, J., Chen, H. and Zhang,X. (2021). Learning to Drop: Robust Graph Neural Network via Topological Denoisings. *WSDM '21: The Fourteenth ACM International Conference on Web Search and Data Mining.* ACM.
[23] Mikolov, T., Chen, K., Corrado, G. and **Dean, J.** (2013). Efficient estimation of word representations in vector space. *ICLR.*
[24] Nghiem, L., Hui, F., Muller, S. and Welsh, A. (2022). Estimation of graphical models for skew continuous data. *Scandinavian Journal of Statistics.* DOI 10.1111/sjos.12569.
[25] Ning, Y. and Liu, H. (2013). High-dimensional semiparametric bigraphical models. *Biometrika* **3.3(2013)** 655–670. MR3094443
[26] Pan, L., Li, C., Li, J. and **Jie, T.** (2017). Prerequisite Relation Learning for Concepts in MOOCs. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics(Volume 1: Long Papers).*
[27] Pennington, J., Socher, R. and Manning, C. (2014). Glove: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* 1532–1543. MR3382218
[28] Racine, J.S. (2008). *Nonparametric Econometrics: A Primer.* Quantile. MR2283034
[29] Rao, C. R. (1973). *Linear statistical inference and its applications, 2nd edition* Wiley, New York. MR221616
[30] Ross, S. M. (2009). *A first course in probability, 8th edition* Pearson Prentice Hall. MR0380910
[31] Wang, J., Yue, K., Duan, L., Qi, Z. and Qiao, S. (2022). An efficient approach for multiple probabilistic inferences with Deepwalk based Bayesian network embedding. *Knowledge-based Systems* **239** 107996.
[32] Yin, J. and Li, H. (2012). Model selection and estimation in matrix normal graphical model. *Journal of Multivariate Analysis* **107** 119–140. MR2890437
[33] Zhang, X. and Zitnik, M. (2020). GNNGuard: Defending Graph Neural Networks against Adversarial Attacks. *Proceedings of Neural Information Processing Systems, NeurIPS.* ACM.
[34] Zhou, J., Li, Y., Zheng, Z. and Li, D. (2022). Reproducible learning in large-scale graphical models. *Journal of Multivariate Analysis* **189** 104934. MR4355938
[35] Zhu, D., Zhang, Z., Cui, P. and Zhu, W. (2019). Robust Graph Convolutional Networks Against Adversarial Attacks. *the 25th ACM SIGKDD International Conference.* ACM.

Jizheng Lai
Center for Applied Statistics and School of Statistics
Renmin University of China
Beijing 100872 China
E-mail address: rucljz@163.com

Jianxin Yin
Center for Applied Statistics and School of Statistics
Renmin University of China
Beijing 100872
China
E-mail address: jyin@ruc.edu.cn
url: http://stat.ruc.edu.cn/jxtd/jsdw/sjkxydsjtjx/5f21fa843de548c8a197c861defc10d2.htm