Application of Graph Auto-encoders Based on Regularization in Recommendation Algorithms

Chengxin Xie [1], **Xiumei Wen** [Corresp., 1, 2], **Hui Pang** [Corresp., 1], **Bo Zhang** [1]

[1] College of Information Engineering, Hebei University of Architecture, Zhangjiakou, China

[2] Big Data Technology Innovation Center of Zhangjiakou, Zhangjiakou, China

Corresponding Authors: Xiumei Wen, Hui Pang
Email address: xiumeiwen@163.com, 23283162@qq.com

Social networking has become a hot topic, in which recommendation algorithms are the most important. Recently, the combination of deep learning and recommendation algorithms has attracted considerable attention. The integration of autoencoders and graph convolutional neural networks, while providing an effective solution to the shortcomings of traditional algorithms, fails to take into account user preferences and risks over-smoothing as the number of encoder layers increases. Therefore, we introduce L1 and L2 regularization techniques and fuse them linearly to address user preferences and over-smoothing. In addition, the presence of a large amount of noisy data in the graph data has an impact on feature extraction. To our best knowledge, most existing models do not account for noise and address the problem of noisy data in graph data. Thus we introduce the idea of denoising autoencoders into graph autoencoders, which can effectively address the noise problem. We demonstrate the capability of the proposed model on four widely used datasets and experimentally demonstrate that our model is more competitive by improving up to 1.3,1.4, and 1.2 respectively on the edge prediction task.

1 Application of Graph Auto-encoders Based on
2 Regularization in Recommendation Algorithms

3 Chengxin Xie[1], Xiumei Wen[12], Hui Pang[1], Bo Zhang[1]

4 [1] College of Information Engineering, Hebei University of Architecture, Zhangjiakou, Hebei,

5 China.

6 [2]Big Data Technology Innovation Center of Zhangjiakou, Zhangjiakou, Hebei, China

7 Corresponding Author:

8 Xiumei Wen, Hui Pang

9 NanzhanJiedao, Zhangjiakou, Hebei, 075000, China

10 Email address: xiumeiwen@163.com; 23283162@qq.com

12 Abstract

13 Social networking has become a hot topic at the moment, in which recommendation algorithms

14 are the most important. Recently, the combination of deep learning and recommendation

15 algorithms has attracted considerable attention. The combination of autoencoders and graph

16 convolutional neural networks, while providing an important solution to the shortcomings of

17 traditional algorithms, fails to take into account user preferences and risks over-smoothing as the

18 number of encoder layers increases. Therefore, we introduce L1 and L2 regularization techniques

19 and fuse them linearly to address user preferences and over-smoothing. In addition, the presence

20 of a large amount of noisy data in the graph data has an impact on feature extraction. To the best

21 of our knowledge, most existing models do not account for noise, and to address the problem of

22 noisy data in graph data, we introduce the idea of denoising autoencoders into graph

23 autoencoders, which can effectively address the noise problem. We demonstrate the capability of

24 the proposed model on four widely used datasets and experimentally demonstrate that our model

25 is more competitive by improving up to 1.3, 1.4, and 1.2 respectively on the edge prediction task.

26 **Keywords:** Graph auto-encoders; Recommendation algorithm; Noise; Regularization

28 1 Introduction

29 Traditional machine learning cannot handle complex graph data. Deep learning is an

30 improvement and extension of it that enhances the structure of learning algorithms [31,32,39]and

31 has been proven to be quite effective in the study of regular and Euclid data, but it is not possible

32 to investigate non-Euclidean or irregular data [1], such as the hyperlinks existing in worldwide

33 web pages cannot be studied [2]. Non-Euclidean data are often irregular, voluminous, and

34 complex, and dependencies between data are often represented by the graph. That doesn't apply

35 to neural networks that deal with regular data, and there are some challenges. To resolve this

36 problem, the researchers introduced the graph neural network (GNN) [3], which received

37 considerable attention. Kipf and Welling [4] proposed the process of message passing between

38 nodes and neighboring nodes to obtain the entire graph structure, which extends the traditional

39 GNN and applies it to graph data. The broad tasks of graph neural networks are graph

40 embedding, link prediction, and clustering. Recently graph representation learning has received a
41 lot of attention and researchers have carried out extensive research to obtain a better
42 representation of nodes. Jiao et al [30] classified graph representation learning into the non-
43 neural network and neural network approaches and collated relevant applications of graph
44 representation learning in computer vision tasks, Shurrab and Duwairi [33] proposed to combine
45 self-supervised learning and computer vision to be able to better handle computer vision tasks.

46   As an important branch of artificial intelligence, recommendation algorithms have received
47 extensive attention in various fields. The combination of recommendation algorithms and smart
48 cities is an example. Shuying et al [34] used recommendation algorithms to achieve commercial
49 site selection and manage smart cities and societies to maximize the efficient use of limited reso
50 In terms of data representation recommendation algorithms can be divided into matrix
51 representations, feature vectors, and graphical models. Embedded representations of graph nodes
52 are widely used in machine learning-related tasks. Unsupervised learning-based auto-encoders
53 can capture complex relationships between nodes by overlaying multiple non-linear layers, but
54 traditional autoencoders cannot use structured data [10].

55   Link prediction is one of the important research contents of graph neural networks, and it is
56 now widely used in the field of recommendation and the citation of academic papers [1]. The
57 theory of applying convolutional neural networks to graph auto-encoders and variational graph
58 auto-encoders was proposed by Kipf and Welling [4]. It is an unsupervised node embedding
59 method for link prediction on graph structures. One of the most classic applications was
60 proposed by Van den Berg et al [9], The model only applies the graph auto-encoder to the
61 recommendation algorithm and does not take into account the user's preferences and the noise
62 problem present in the graph data. Yan et al [38] proposed a new IoT-based intelligent product
63 recommendation system based on an apriori algorithm and fuzzy logic, capable of determining
64 users' preference questions and filtering irrelevant information. Although the model can
65 determine user preferences to achieve recommendations, it does not consider the problem of
66 noisy data. The graph auto-encoders are applied to recommendation algorithms to complete link
67 prediction [11], but the model did not take into account user preferences [10,41]. In addition, the
68 use of graph convolutional neural networks as encoders also has many disadvantages, as the
69 number of layers of graph convolution increases, the model parameters are difficult to optimize
70 [6,40]. Not only is there a lot of useful data in the graph data, but there is also a lot of noisy data
71 [14], which is not considered by the existing models [1, 6, 9]. Therefore, they cannot extract
72 features well, which has a certain impact on downstream tasks.

73   In this paper, we propose a graph auto-encoder fused with regularization to better address
74 user preferences for items and over-smoothing issues. We introduce a penalty term.
75 Simultaneously, we introduce the idea of denoising auto-encoder into the graph auto-encoders to
76 extract useful information from the noise data to restore the original data. To summarize, the
77 contributions of our paper are as follows:

78   1. To better extract the data in the graph and solve over-smoothing, we fuse the denoising
79 autoencoder and the graph auto-encoders (D-GAE). Adding noise helps the model to better
80 extract useful data for reconstructing the user-item rating matrix.
81   2. To better solve the problem of user preference and the risk of overfitting, we introduce
82 L1(GAE+L1) and L2(GAE+L2) regularization. To fuse the advantages of L1 and L2(GAE+L1+
83 L2), we fuse L1 and L2 regularization linearly to better solve the risk of overfitting.
84   3. We experimented using four datasets of Ml-100k, Flixster, Douban, and YahooMusic.
85 Experiments proved the feasibility and correctness of our ideas. The variance of our proposed
86 algorithm is smaller.

87

88 2 Related Work

89 Low-dimensional representations of nodes in the graph have been proven useful in machine
90 learning, such as node classification, social networking, link prediction, and recommendation
91 systems [5]. Graph embedding is the process of mapping high-dimensional and complex
92 information in a graph to a low-dimensional dense vector, and effectively realizes the reduction
93 of feature dimensionality [6]. Traditional auto-encoders cannot use display relationships in graph
94 structures. To solve this problem, Shirui et al [7] proposed that graph auto-encoders can use
95 graph structures, but ignore the reconstructed node features or graph structures. Traditional auto
96 encoders decode from low-dimensional representations of the entire receptive field of a neural
97 network. For GNNs, the receptive field represented by a node is its entire neighborhood.
98 However, existing graph auto-encoders only decode the direct links between node pairs by
99 minimizing the link reconstruction loss [6], which results in nodes losing a great deal of
100 information, which affects downstream tasks. In this regard, Salehi and Davulcu [5] proposed a
101 graph attention auto-encoder by stacking the encoder and decoder layers to better reconstruct the
102 node features of the graph structure and regularized the node representation to reconstruct the
103 graph structure.
104   Traditional recommendation algorithms have data sparsity and cold-start problems [8]. To
105 solve this problem, a graph neural network recommendation algorithm is proposed, which solves
106 the problems existing in traditional recommendation algorithms [9]. But the algorithm does not
107 consider user preferences and the possible risk of overfitting. Xingjie and Xiaoyu [10] pointed
108 out that most graph neural network recommendations ignore the preference problem of user
109 product reviews, and proposed an item recommendation algorithm combining graph neural
110 network and deep learning for this problem. This algorithm solves the problem of ignoring user
111 product reviews in most graph neural network recommendations and combines deep learning to
112 extract user preferences.
113   There are not only many useful data in the graph data, but also a lot of noise [8, 14, 15,28],
114 and it is also pointed out that there is an over-smoothing problem in the graph neural network
115 when there are too many layers. Recently Salha et al [11] pointed out that auto-encoders based
116 on graph convolution use two or three layers of shallow structure. When the number of layers is
117 large, the ability of link prediction cannot be improved, and there is a problem of over-

118 smoothing. Aiming at these two problems, a bidirectional collaborative filtering algorithm based
119 on graph convolution is proposed. The algorithm solves the over-smoothing problem of graph
120 neural network and noise during propagation, divides the graph into several subgraphs for
121 learning, and uses an attention mechanism to optimize the representation of several nodes.
122 Because of a large amount of noisy data [8,13], to improve the robustness of the model [13], an
123 enhanced model of variational auto-encoder is proposed to obtain a noise-resistant
124 recommendation model. The proposed model is robust to noise, and a hidden layer is added to
125 the variational autoencoder to obtain a noise-resistant recommendation model.
126   Our work is based on the above-mentioned noise interference and user preference problems.
127 The vast majority of models do not take into account the problem of errors arising from user
128 preferences, so we propose a regularization technique to eliminate user preferences. Existing
129 models do not take into account the fact that graph data is also noisy and interferes with feature
130 extraction. Therefore, we introduce the idea of denoising auto-encoders into graph auto-
131 encoders, which can solve the interference of noise and better restore the original data.
132

133 ## 3 Methodology
134 Chapter 3 focuses on the relevant techniques used by the proposed model.
135 ### 3.1 Graph Convolutional Neural Networks
136 Graph Convolutional Neural Networks (GCN) can extract high-dimensional image features,
137 which can be divided into the frequency domain and spatial domain GCN [26]. GCN aggregates
138 the information of the node itself and surrounding neighboring nodes to obtain the node
139 representation of the node in the next layer [22]. The specific definition is shown in the
140 following formula (1):

141 $H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}w^{(l)})$[4]   (1)

142 Where $\tilde{A}=A+E$, A is the adjacency matrix, and E is the identity matrix. D is the degree matrix,

143 $\tilde{D}^{-\frac{1}{2}}$ is the normalization operation, $H^{(l)}$ represents the characteristics of nodes, $w^{(l)}$ is the weight
144 matrix and $H^{(l+1)}$ is the characteristics of nodes in the next layer.
145

146 ### 3.2 Bipartite graph
147 The recommendation is based on the users and items. Assuming that users and items only have
148 ratings and no other relationship, users, and items can form a bipartite graph [9], as shown in Fig.
149 1. A rating matrix of user-item of size M *N can be constructed from the bipartite graph. r_{ij}
150 represents the user i's rating for item j, and the weight coefficient of the edge is r_{ij}. When r_{ij} is 0,
151 it means that user I have no rating for item j. The larger the value of r_{ij}, the more interested the
152 user is in the item [1,2]. The elements in the rating matrix are the user's rating. The
153 implementation of user recommendations for items is essentially the problem of edge prediction,
154 that is, the rating matrix that fills the vacancies [9].
155

156 ### 3.3 Regularized Graph Auto-encoders Recommendation Algorithm

157 Graph auto-encoders (GAE) can implement link prediction tasks. The essence of GAE is to
158 implement a graph convolutional neural network (GCN) on an auto-encoder [29]. Similar to
159 auto-encoders, GAE can reconstruct the adjacency relationship between nodes [19]. Salehi et al
160 [5] proposed an attention-based graph auto-encoders that uses a self-attention mechanism to
161 learn the embedding representations of nodes in graph data unsupervised. Auto-encoders
162 reconstruct the graph structure and node features by stacking encoder or decoder layers. The
163 same is that the graph auto-encoders can be divided into two processes: encoding and decoding.
164 Unlike auto-encoders, GAE's encoding process uses convolutions.
165   1. Encoder
166   GAE adopts a GCN as the encoder to obtain the embedding representation of nodes. The
167   encoder uses node features and adjacency matrices and generates embedding representations of
168   nodes by stacking [20,29]. The specific definition is shown in the following formula (2):
169    $Z = GCN(X, A)$ [36]   (2)
170   Where Z represents the embedding representation of all nodes, X represents the feature
171 matrix of nodes, and A represents the adjacency matrix. Use GCN to take X and A as inputs to
172 obtain output Z.
173   The specific definition of GCN is shown in the formula (3):
174    $GCN(X, A) = \tilde{A} \, Relu(\tilde{A} X w_0) w_1$ [36]   (3)

175   Where $\tilde{A} = D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$. D is the degree matrix, w_0、 w_1 is a learnable parameter. Here, the
176 GCN is equivalent to the input node feature and adjacency matrix to obtain the function of the
177 embedding representation of the node.
178   2. Decoder
179   GAE uses an inner product to represent the adjacency relationship of nodes and reconstructs
180 the original adjacency matrices [29]. The specific definition is shown in formula (4):
181    $\hat{A} = \sigma(ZZ^T)$ [36]   (4)

182   Where $\hat{A}$ is the reconstructed adjacency matrix, σ is the activation function.
183   Using the graph auto-encoders to achieve the error between the real rating and the predicted
184 rating, that is, use the loss function to train and reconstruct the rating matrix. A GCN is used as
185 the encoder to obtain the embedding representation of each node, and the original adjacency
186 matrix is reconstructed by the inner product to complete the edge prediction [9]. The overall
187 structure is shown in Fig.2. The regularized graph auto-encoder recommendation algorithm uses
188 the feature matrix X and the rating matrix of user-item as input, and generates node embedding
189 representation. The decoder model uses the node embedding representation to reconstruct a
190 matrix of user and item ratings [9]. Graph auto-encoders can implement weight sharing on the
191 graph and separately assign processing channels for the type of each edge or rating level R.
192 Weight sharing is a characteristic of graph convolutional neural networks. Its implementation is
193 derived from convolutional neural networks, which operate directly on graph-structured data
194 [21].

195   During training, the rating matrix of the user and item is normalized. The graph
196   convolutional encoder uses the relu activation function to obtain the hidden layer features of
197   users and items. The related features of users and items are normalized by the softmax function
198   to obtain the probability of users and items features, we use two GCN layers to non-linearly
199   transform the hidden layer features of the user-item and its features by stack [9] to obtain the
200   feature expression of the user-item. The cross-entropy loss function is used to adjust the model to
201   improve its performance of the model. As the number of layers of the GCN model increases, it
202   will lead to the over-smoothing problem [1], which makes the reconstructed adjacency matrix
203   tends to be consistent with the original matrix, and the parameters are difficult to optimize.
204   Moreover, errors are caused by user preferences. Therefore, we introduce L1 regularization and
205   L2 regularization penalty terms. The L1 regularization term is the sum of the absolute values of
206   the elements of the weight vector w. It is used to perform a sparse operation so that the model
207   cannot fit arbitrary data. The L2 regularization term is the sum of the squares of the elements of
208   the weight vector w, which prevents the overfitting of the model and enhances the model's ability
209   to resist interference. To better extract features, we linearly fuse the L1 and L2 regularization
210   penalty terms to enhance the generalization ability of the model. This solves the overfitting
211   problem.
212   The specific definition is shown in Formula (5) after adding the L1 regularization penalty
213   term based on the loss function.

214    $$L(y, f(x)) = -\sum_{i,j\Omega_{ij}=1}\sum_{r=1}^{R}I(r = A_{ij})\log P(\hat{A}_{ij} = r) + \lambda(|w_u|) \qquad u=1,2 \quad (5)$$

215   The specific definition is shown in Formula (6) after adding the L2 regularization penalty
216   term based on the loss function.

217    $$L(y, f(x)) = -\sum_{i,j\Omega_{ij}=1}\sum_{r=1}^{R}I(r = A_{ij})\log P(\hat{A}_{ij} = r) + \lambda(\|w_u\|_2) \quad u=1,2 \quad (6)$$

218   Based on the loss function, the L1 and L2 regularization penalty terms of linear fusion are
219   added, and the specific definition is shown in formula (7):

220    $$L(y, f(x)) = -\sum_{i,j\Omega_{ij}=1}\sum_{r=1}^{R}I(r = A_{ij})\log P(\hat{A}_{ij} = r) + \lambda(|w_u| + \|w_v\|_2) \quad u,v=1,2 \quad (7)$$

221   where I[k=l] = 1 when k = l and zero otherwise. The matrix $\Omega \in \{0, 1\}$ denotes the mask
222   without a rating, if the score is 1 then the corresponding element in the rating matrix is 1
223   otherwise it is 0. A_{ij} denotes the rating matrix for the user and item. $\hat{A}_{ij}$ denotes the rating matrix
224   after the reconstruction of user i and item j. λ is the penalty factor and w is the weight matrix.
225   In addition to the above problems, there is still noise in the graph, which causes certain
226   interference to feature extraction [8]. As the number of GCN layers increases, there will be an
227   over-smoothing problem, which makes the embedding representation of nodes tend to be
228   consistent and the model difficult to optimize. To better extract features and solve existing
229   problems, we combine denoising auto-encoders and graph auto-encoders, adding 20% noisy data
230   to the rating matrix of user-item to better extract useful data. The effectiveness of our idea is

231   demonstrated by multiple experiments on four datasets. The experimental results show that the
232   effect of adding 20% noise data is performance.

233

234   **3.4 Model flow chart**
235   The regularized graph auto-encoder recommendation algorithm shown in Fig.3 mainly includes
236   the following processes:

1.   Data loading. In this step, our model takes the feature matrix X and the rating matrix of the user and item as input. On this basis, we introduce the rating matrix into the noise data.
2.   Encoding. The initial features of the user and the item are first fed into the GCN layer to obtain a hidden layer representation of the user and the item. The edge information between the user and the item is then fed into the fully connected layer to obtain the hidden layer representation of the edge. The initial user and item hidden layer features and the hidden layer features of the edges are then fed into the fully connected layer to obtain the embedded representations of the user and the item, completing the encoding process.
3.   Decoding. GAE uses an inner product to reconstruct the rating matrix. The model is optimized using the cross-entropy loss function, and the parameters are optimized by gradient descent. Based on loss function, we introduce L1 and L2 regularization techniques and linearly fuse them.
4.   Results. Finally, the training loss and root mean square error (RMSE) in the training process of the model is output.

251

252

253   # 4 Experimental Results and Analysis

254   Chapter 4 focuses on the model parameters and the experimental results.
255   **4.1 Datasets**
256   In this paper, we have used four datasets: ML-100k, Flixster, Douban, and YahooMusic [25].
257   These datasets contain user and item information, and feature vectors composed of users and
258   items are normalized using a rating matrix. We separate the 80/20 training/validation set from
259   the original training set, which helps us evaluate the final model performance. We train on the
260   training set and output the experimental results on the test set [9]. The overall structure is shown
261   in Fig.4.

262

263   **4.2 Evaluating index**
264   The evaluation index used in this paper is the root mean square error. RMSE can measure the
265   error between the real value and the predicted value. The specific definition is shown in the
266   formula (8):

267    $$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(h\!\left(x^{(i)}\right) - y^{(i)}\right)^2}\ [9] \qquad\qquad (8)$$

268   Where N represents the total number of scores, $h(x^{(i)})$ represents the predicted value, and
269    $y^{(i)}$ represents the real value. The smaller the error value, the better the model performance.

270

271 **4.3 Algorithm flow**

Algorithm 1   Regularized Graph Auto-encoders Recommendation Algorithm

Inputs: Graph (V, X, A), name of dataset d, discard rate p, the aggregation approach s, training times e, noise n.

1: Begin

2: Add n $\longrightarrow$ A (Rating matrix for users and items), and input X (the feature matrix of users and items)

3: For i=0 to e do

4:    Normalize the Rating matrix and input user features and item features.

5:    Input the results obtained in step 4 into the GCN layer.

6:   Calculate the user's hidden layer representation using equations (1), (2), and (3).

7:   Calculate the item's hidden layer representation using equations (1), (2), and (3).

8:    Add bias items to steps 6 and 7 and output user and item features.

9 :   Input the user's edges and the item's edges and use the fully connected layer to obtain the features of the user's edges and the features of the item's edges.

10.   Concatenate step 9 with user features and item features to obtain item features and user features.

11.   The result obtained in step ten is fed into the fully connected layer to obtain an embedded representation of the user and the item.

12:    Reconstruct the rating matrix using equation (4).

13**:**   Optimization of the model using equation (5) and optimization of the parameters by gradient descent.

14**:**   Optimization of the model using equation (6) and optimization of the parameters by gradient descent.

15**:**   Optimization of the model using equation (7) and optimization of the parameters by gradient descent.

16: END For

17**:** Output the training loss of the model and calculate the RMSE using equation (8).

18: END

272

273 **4.4 Experiment Setup**

274 **Experiment settings**. The evaluation index used in our experiments is RMSE [9]. All

275 hyperparameters in the paper remain the same as in [9], and none of them have been modified.

276 This has the advantage of better illustrating the validity of our model under equivalent

277 circumstances. The total number of epochs of our experiments in the movie dataset ml-100k is

278 2000, and the initial learning rate of the optimizer is 0.001. The input dimension of the node is

279 2625, and the GCN is hidden. The dimension of the layer is 500, the hidden dimension of the

280 encoder is 75, the bias is 2, and the dropout is 0.5. Additionally, our evaluation metric on the

281 Flixster, Douban, and YahooMusic datasets is also RMSE, and all parameters are the same. The
282 total number of experiments on these three datasets is 200 epochs, and the initial learning rate is
283 0.01. We have continued all the parameter conditions of [9] to better illustrate the experimental
284 results, hidden= [500,75], dropout is 0.7, bias is 2, feat-hidden=64. Additionally, we added 20%
285 noise data based on the experiments of the four datasets to better obtain the data in the graph and
286 complete the reconstruction.

287 **Result on Ml-100k**. The initial user input feature is [943,2625], the item input feature is
288 [1682,2625], the input feature dimension is 2625, and the output feature dimension is 500. User-
289 item adjacency matrix corresponding to each rating level after normalization is (943, 1682). The
290 item-user adjacency matrix corresponding to each rating level after normalization is (1682,943).
291 The normalized two matrices, the user's input, and the item's input are passed through the GCN
292 layer to obtain a hidden layer representation of the user and the item. After the GCN layer, the
293 user hidden layer feature is [943,500] and the item hidden layer feature is [1682,500]. The user
294 features and item features are then output by adding the bias term to the user features and item
295 features. The user feature is [943,500] and the item feature is [1682,500]. The input to the user
296 edge is [943,41] and the input to the item edge is [1682,41]. The features of the user edge
297 [943,10] and the features of the item edge are obtained after the first hidden layer, based on the
298 input of the user edge and the input of the item edge. The features of the user edge are [943,10]
299 and the features of the item edge are [1682,10]. The user's features are then the features of the
300 concatenated user edges and the user features are obtained using the GCN layer. The user's
301 features are [943,510]. Similarly, the item features are the features of the concatenated item
302 edges and the item features obtained using the GCN layer. The item features are [1682,510].
303 Once the user's features and the item's features are obtained, the embedding representation of the
304 user and the item can be obtained after the second hidden layer of the reddest. The embedding
305 representation of the user is [943,75] and the embedding representation of the item is [1682,75].
306 The encoding process is then completed. The final edge prediction is [100000,5] obtained by the
307 decoder using the embedded representation of the user and the item.

308 We summarize the RMSE in our experimental results as shown in Fig.5. We compared
309 several other classic models with our proposed optimization, and we can see that our proposed
310 model has better performance and is better. After many experiments, the effect of the graph auto-
311 encoders with L1 and L2 regularization is improved by 0.2%. In addition, we also recorded the
312 variation trend of Loss and RMSE in the process of model training 2,000 times. The specific
313 experimental results can be seen in Fig. 6 and Fig.7.

314 **Results on Flixster, Douban, and YahooMusic**. To verify the usability of our proposed
315 idea, we verified it in another three datasets and proved the correctness of our idea through
316 experiments. We run multiple averages on the three datasets [9], and the experiments show that
317 GAE+L1/L2 optimizes by 0.6%, 0.5%, and 1.3 on the three datasets, respectively, relative to the
318 GC-MC algorithm. A similar fused L1 and L2 regularization graph auto-encoders GAE+L1+L2
319 is optimized by 1%, 0.6%, and 1.4 on the three datasets, respectively. Also, fused denoising
320 autoencoder and graph auto-encoders were optimized by 0.3% and 1.2 on three datasets. The

321 specific experimental results are shown in Fig. 8. As can be seen from Figure 8, in the three data
322 sets comparing the GAE+L1+L2 and GAE+L1 models, GAE+L1+L2 improves by 0.4%, 0.2%,
323 and 0.1. However, comparing the GAE+L1+L2 and GAE+L2 models, GAE+L1+L2 improves by
324 0.1%, and 0.2. As can be seen from the above two comparisons, fusing L1 and L2 is more
325 effective than either L1 or L2 are more effective. After many experiments on the three datasets,
326 we found that the variance and error of our algorithm are smaller, and it does not affect the time
327 complexity.
328   The experimental results on the four datasets show that our idea is valid and correct, and the
329 best GAE+L1+L2 model has an RMSE of 19.1 on the Yahoo dataset which is the best available
330 model. The D-GAE model has an RMSE of 19.3 on the Yahoo dataset, which is a good solution
331 to the problem of noise interference in the graph data, From the ablation experiment we can tell
332 that the noise figure is relatively good with the addition of 20%. The specific experimental
333 results are shown in Fig. 9. Although we were able to solve some of the user preference and
334 noise interference problems using regularization techniques, there are some limitations to using
335 graph auto-encoder in combination with recommendation algorithms such as the low embedding
336 problem, which we discuss in the conclusion and future directions of work.
337

5 Discussion

339 To solve the problem that traditional recommendation algorithms must have nearby users to
340 achieve recommendations, Cheng and Gao [37] proposed a recommendation algorithm based on
341 deep neural networks. However, the model does not consider the user's preference problem and
342 has certain drawbacks. Van den Berg et al [9] proposed a graph auto-encoder recommendation
343 algorithm that can solve the traditional recommendation algorithm cold start and data sparsity
344 problems but the algorithm does not take into account the interference caused by the user's
345 preferences. There is a lot of noisy data in graph data, and Haiqian et al [13] proposed a noise-
346 resistant recommendation model. However, this model only adds one hidden layer, and as the
347 number of layers increases the parameters for model training increase, the risk of overfitting is
348 too great and the parameters are difficult to optimize. Unlike existing models, our model takes
349 full account of the error problem caused by user preferences, and we have used regularization
350 techniques to achieve good results on all four datasets, as can be seen in the multiple model
351 comparison results. The problem of noise in graph data and how to solve the noise interference
352 for better recommendations is the focus of current consideration. As far as we know, most of the
353 existing models do not take into account noise interference, our D-GAE model can solve this
354 problem very well, and the results are better, and the robustness of the model is better. Our
355 model can be used not only for recommendation problems but also, for example, for the
356 combination of graph auto-encoders and computer vision tasks.

6 Conclusion

358 We introduce L1 and L2 regularization, which can better improve the user's preference problem
359 and over-smoothing problem. The RMSE for the four datasets Ml-100k, Flixster, Douban, and
360 YahooMusic is 0.909, 0.910, 0.730, and 19.2 for the L1 regularization. The RMSEs on the Ml-

361 100k, Flixster, Douban, and YahooMusic datasets were 0.909, 0.910, 0.729, and 19.3
362 respectively when L2 regularization was added. Simultaneously, the linear fusion of L1 and L2
363 regularization can better extract features and prevent overfitting. The RMSE on the Ml-100k,
364 Flixster, Douban, and YahooMusic datasets were 0.908, 0.907, 0.729, and 19.1 respectively. To
365 better extract data features in the graph, we introduce the concept of denoising auto-encoders
366 into graph auto-encoders. Through experiments on four datasets of Flixster, Douban, and
367 YahooMusic, the RMSE can reach 0.916, 0.917, 0.730, and 19.3, respectively. The correctness
368 and effectiveness of our proposed ideas are verified by multiple experiments on four public
369 datasets. The combination of graph auto-encoder and recommendation algorithm, but the graph
370 auto-encoder itself has certain shortcomings simply using the GCN as an encoder is not good for
371 feature extraction, in the future we can consider the combination of random walk graph auto-
372 encoder and recommendation algorithm, the method uses the random walkway to reconstruct the
373 new node embedding matrix, can be better for feature extraction. The random walk-based graph
374 auto-encoder recommendation algorithm is a new approach, and it is also possible to combine
375 other optimized graph auto-encoders with recommendation algorithms to construct new
376 recommendation algorithms which will open another door and have a profound impact on the
377 development of recommendation algorithms. Additionally, although we have proposed a
378 regularization technique, more optimization directions in the future are still worth exploring.
379

7 Acknowledgments

381 My greatest gratitude goes to Pro. Xiumei Wen, my supervisor, for her guidance for me to
382 complete this paper. This paper is based on the research project XY2023020 and XY2023026 on
383 basic scientific research business expenses of provincial colleges and universities in Hebei
384 Province.
385

Reference

387 [1] Xinxing Wu, Qiang Cheng. Stabilizing and Enhancing Link Prediction by Deepened Graph
388 Auto-Encoders.In IJCAI,2022.
389 [2] Hongbin Pei, Bingzhe Wei, Yu Lei, and Bo Yang. Geom-GCN: Geometric graph
390 convolutional networks. In ICLR,2020.
391 [3] Franco Scarselli, Marco Gori, Markus Hagenbuchner, and Gabriele Monfardini. The graph
392 neural network model. IEEE Trans. Neural. Network., 20(1):61–80, 2009.
393 [4] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional
394 networks. In ICLR, 2017.
395 [5] Amin Salehi, Hasan Davulcu. Graph Attention Auto-Encoders.In ICLR,2019.
396 [6] Mingyue Tang, Carl Yang, Pan Li. Graph auto-encoder via neighborhood Wasserstein
397 reconstruction. In ICLR, 2022.
398 [7] Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang.2018.
399 Adversarially Regularized Graph auto-encoders for Graph Embedding. InIJCAI. 2609–2615.

400   [8]Fei Gao, Kaijie Lin .Bidirectional collaborative filtering recommendation algorithm based on
401   graph convolution network [J] Software, 2021 (07): 032-038.
402   [9] Rianne van den Berg, Thomas N. Kipf, Max Welling. Graph Convolutional Matrix
403   Completion. ICLR (2018).
404   [10] Xingjie Feng, Xiaoyu Sheng. Item recommendation algorithm based on graph neural
405   network and deep learning [J] Computer application research, 2021 (12): 016-022.
406   [11] Guillaume Salha, Romain Hennequin, and Michalis Vazirgiannis. Simple and effective
407   graph auto-encoders with the one-hop linear model. In ECML PKDD,2020.
408   [12] David K. Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell,Timothy
409   Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. Convolutional networks on graph for learning
410   molecular fingerprints. In Advances in neural information processing systems (NIPS). 2018.
411   2224–2232.
412   [13]Haiqian Chen, Fu Lan, Jie Chen, Shu Zhao, Yanping Zhang. Scoring prediction model based
413   on variational auto-encoder [J] Computer engineering and application, 2021 (22): 153-159.
414   [14] Edward Turner. Graph auto-encoders for financial clustering. 2021. arXiv preprint
415   arXiv:2111.13519.
416   [15] Francesco Rundo, Francesca Trenta, Agatino Luigi di Stallo, and Sebastiano Battiato.
417   Machine learning for quantitative finance applications: A survey. Applied Sciences, 2019,9(24).
418   [16] Luyu Gao and Jamie Callan. Condenser: a pre-training architecture for dense retrieval.
419   2021.arXiv preprint arXiv:2104.08253.
420   [17] Yun sang Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin
421   Soljacic, Shang Wen Li, Wen tau Yih, Yoon Kim, and James R.Glass. Diffuse: Difference-based
422   contrastive learning for sentence embeddings. In CoRR,2022, abs/2204.10298.
423   [18] Zheng Liu, Yingxia Shao. RetroMAE: Pre-training Retrieval-oriented Transformers via
424   Masked Auto-Encoder. 2022. arXiv preprint arXiv:2205.12035.
425   [19] Xuelong Li, Hongyuan Zhang, and Rui Zhang. Adaptive Graph Auto-Encoder for General
426   Data Clustering. 2022.arXiv preprint arXiv:2002.08648.
427   [20] Sami Abu-El-Haija, Bryan Perozzi, Rami Al-Rfou, and Alex Alemi. Watch Your Step:
428   Learning Node Embeddings via Graph Attention. In NIPS 2018.
429   [21] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional
430   networks. In ICLR,2017.
431   [22] Mingyuan Ma, Sen Na, and Hongyu Wang. AEGCN: An Autoencoder-Constrained Graph
432   Convolutional Network. 2021. arXiv preprint arXiv:2007.03424.
433   [23] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and
434   Max Welling. Modeling Relational Data with Graph Convolutional Networks. In The Semantic
435   Web, Aldo Gangemi, Roberto Navigli, Maria Esther Vidal, Pascal Hitzler, Raphaël Troncy,
436   Laura Hollink, Anna Tordai, and Mehwish Alam (Eds.).Springer International Publishing, Cham,
437   2018,593–607.
438   [24] Alessandro Generale, Till Blume,and Michael Cochez. Scaling R-GCN Training with Graph
439   Summarization. In ACM,2022.

440   [25] Piji Li. An Empirical Investigation of Pre-Trained Transformer Language Models for Open-
441   Domain Dialogue Generation. 2020.arXiv preprint arXiv:2003.04195.

442   [26] Hongbin Pei, Bingzhe Wei, Kevin Chenchuan Chang, Yu Lei, and Bo Yang. Geom-GCN:
443   Geo-metric graph convolutional networks. In ICLR, 2020.

444   [27] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep
445   graph convolutional networks. In ICML, 2020.

446   [28] Wenzheng Feng, Jie Zhang, Yuxiao Dong, Yu Han, Huanbo Luan, Qian Xu, Qiang Yang,
447   Evgeny Kharlamov, Jie Tang. Graph Random Neural Networks for Semi-Supervised Learning
448   on Graph.In NeurIPS, 2020.

449   [29] Guillaume Salha, Romain Hennequin, Manuel Moussallam, Michalis Vazirgiannis,
450   FastGAE: Scalable graph auto-encoders with stochastic subgraph decoding, Neural
451   Networks,21(142)1-19,2021.https://doi.org/10.1016/j.neunet.2021.04.015.

452   [30] L. Jiao et al., Graph Representation Learning Meets Computer Vision: A Survey, in IEEE
453   Transactions on Artificial Intelligence, 2022, DOI: 10.1109/TAI.2022.3194869.

454   [31]Ah A , Njn B , Mu C , et al. The COVID-19 epidemic analysis and diagnosis using deep
455   learning: A systematic literature review and future directions. 2021.

456   [32] Heidari, A., Jafari Navimipour, N., Unal, M. et al. Machine learning applications for
457   COVID-19 outbreak management. Neural Comput & Applic 34, 15313–15348 (2022).

458   [33] Shurrab S, Duwairi R. Self-supervised learning methods and applications in medical
459   imaging analysis: a survey. PeerJ Computer Science 8:e1045, 2022.https://doi.org/10.7717/peerj-
460   cs.1045.

461   [34] Zhai Shuying, Hao Shaoyang, Yang Qi, et al. Intelligent business location recommendation
462   algorithm based on multi-source heterogeneous data fusion [J]. Modern Electronic Technology,
463   2019, 42(14):5.

464   [35] Heidari, Arash, Nima JafariNavimipour, MehmetUnal. Applications of ML/DL in the
465   management of smart cities and societies based on new trends in information technologies: A
466   systematic literature review[J]. Sustainable Cities and Society. 2022.

467   [36] T. N Kipf and M. Welling. Variational graph auto-encoders. In NIPS, 2016.

468   [37] Cheng L, Gao M T. Recommendation Algorithm Based on Deep Neural Networks[J].
469   Modern Computer, 2018.

470   [38] S. -R. Yan, S. Pirooznia, A. Heidari, N. J. Navimipour and M. Unal.Implementation of a
471   Product-Recommender System in an IoT-Based Smart Shopping Using Fuzzy Logic and Apriori
472   Algorithm[J] in *IEEE Transactions on Engineering Management*, 2022, doi:
473   10.1109/TEM.2022.3207326.

474   [39] Arash Heidari, Nima Jafari Navimipour, Mehmet Unal, Guodao Zhang.Machine Learning
475   Applications in Internet-of-Drones: Systematic Review, Recent Deployments, and Open
476   Issues[J] ACM Computing Surveys,2022,doi.org/10.1145/3571728.

477   [40] Salha G, Hennequin R, Remy J B, et al. FastGAE: Scalable Graph Autoencoders with
478   Stochastic Subgraph Decoding[J]. Neural Networks, 2021.

479   [41] TIANWEI WANG. Personalized Movie Recommendation Algorithm Based on Knowledge
480   Graph[D]. Hubei: Huazhong Normal University,2020.
481

Figure 1

Rating Matrix and Bipartite Graph

The figure 1 describe the Rating Matrix and Bipartite Graph
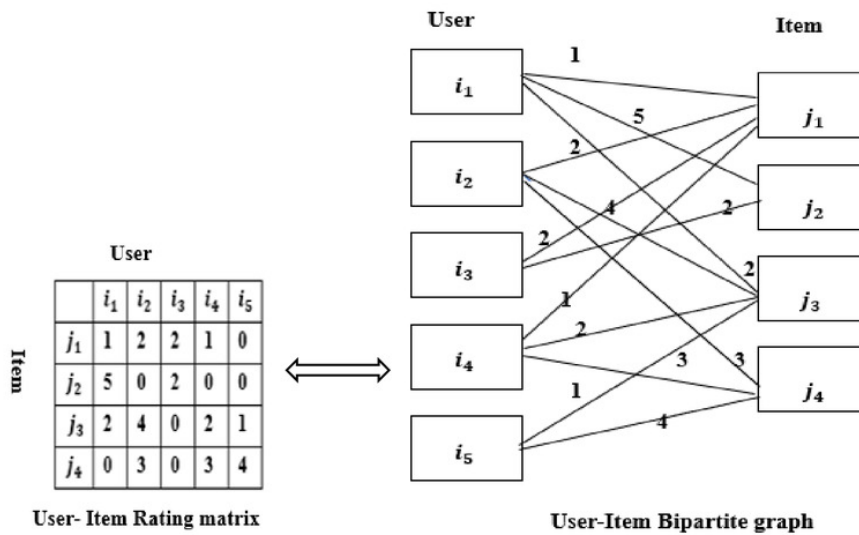
Figure 1. Rating Matrix and Bipartite Graph

Figure 2

Graph Auto-Encoders Structure Graph

The figure 2 describe the Graph Auto-Encoders Structure Graph
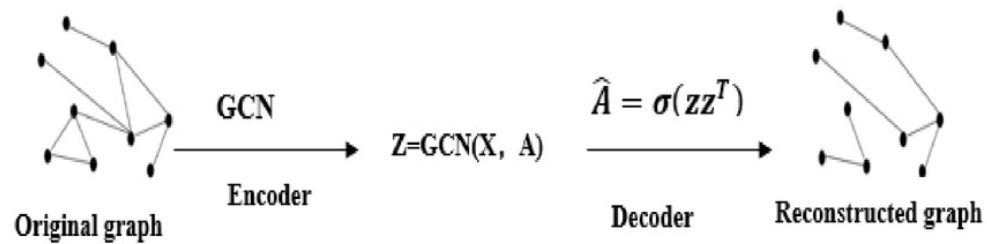
Figure 2. Graph Auto-Encoders Structure Graph

Figure 3

Model flow chart

Figure 3 describe the overall flow chart of the experiment
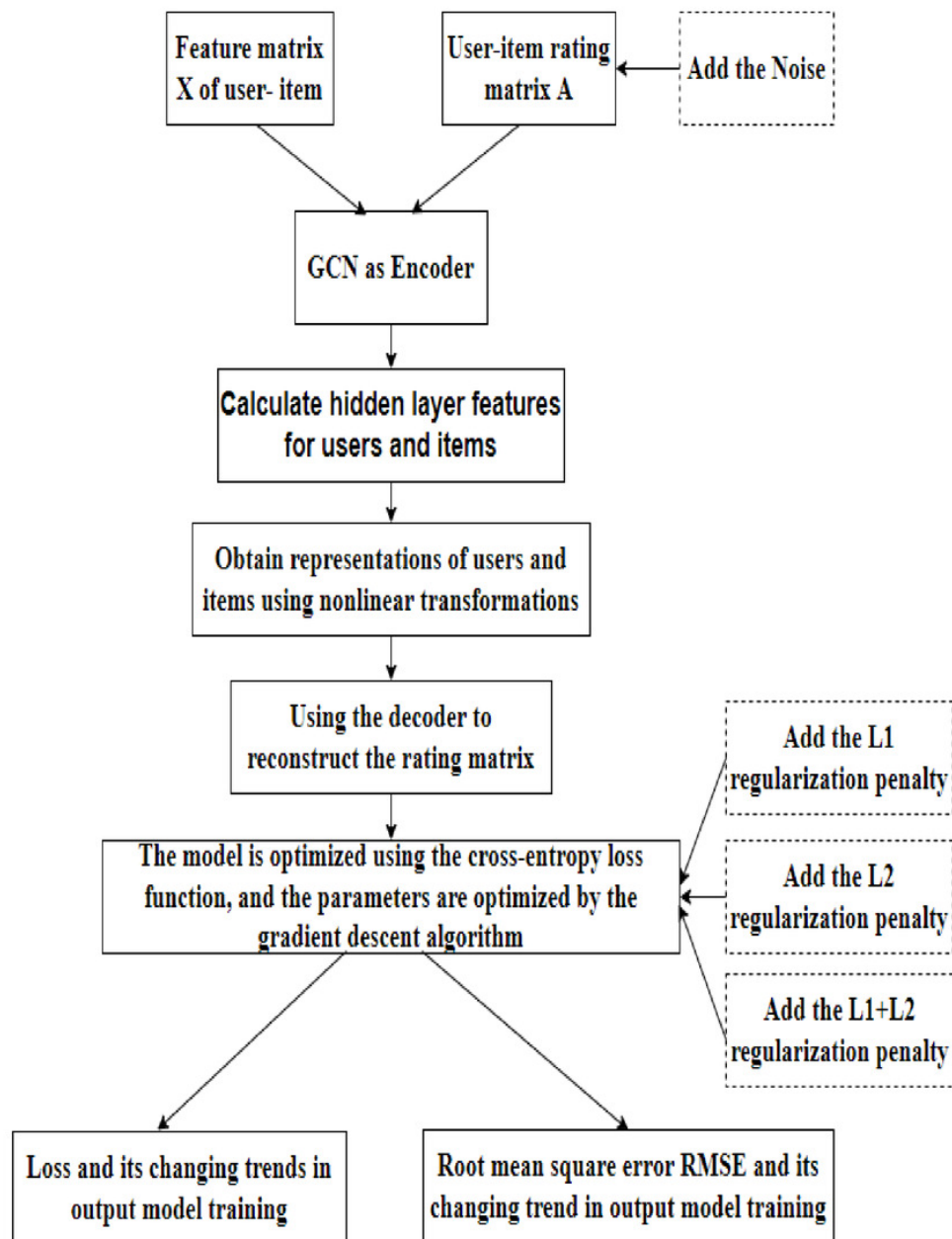
Figure 3. Model flow chart.

Figure 4

Four datasets

Figure 4 describe the four datasets used in the experiment

Dataset	Users	Items	Features	Ratings	Density	Ratinglevels
ML-100k	943	1682	Users/Items	100000	0.0630	1-5
Flixster	3000	3000	Users/Items	26173	0.029	0.5,1,....5
Douban	3000	3000	Users	136891	0.0125	1-5
YahooMusic	3000	3000	Items	5335	0.0006	1-100

Figure 4. four datasets.

Figure 5

Model comparison

Figure 5 describe the comparison of our model on the Ml-100k dataset with other models.

Model	ML-100K
User-based CF	1.275
Item-based CF	1.119
AE	0.983
SVD	0.944
MC	0.973
IMC	1.653
GMC	0.996
PinSage	0.951
GRALS	0.945
sRGCNN	0.929
GC-MC	0.910
NMTR	0.911
F-EAE	0.920
D-GAE(ours)	0.916
GAE+L1/L2(ours)	0.909
GAE+L1+L2(ours)	0.908

Figure 5. Model comparison.

Figure 6

Experimental results(GAE+L1+L2 basis onMl-100k)

Figure 6 describe the trend of our model(GAE+L1+L2) on the Ml-100k test loss and test RMSE
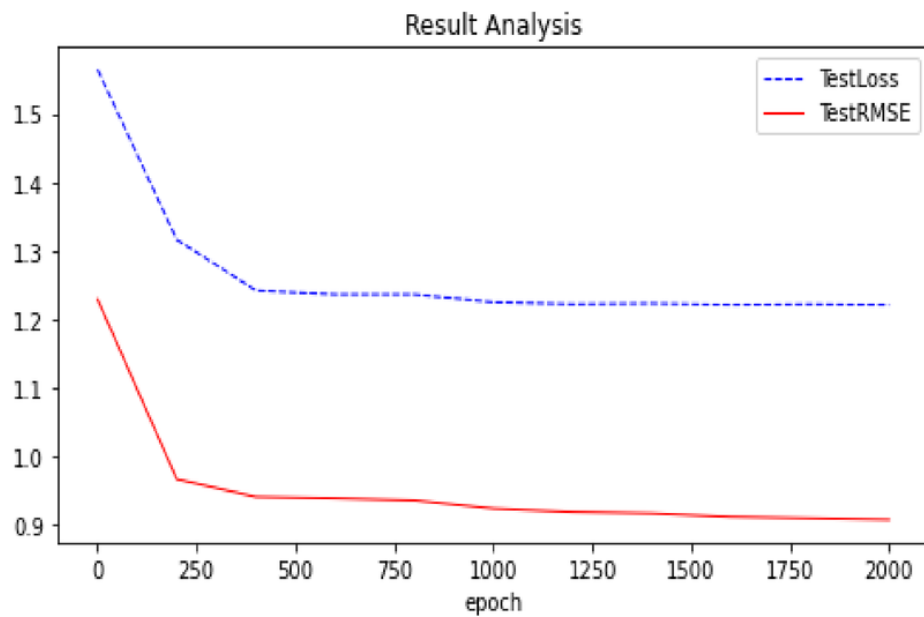
Figure 6. Experimental results (GAE+L1+L2 basis on Ml-100k).

Figure 7

Experimental results(D-GAE basis onMl-100k)

Figure 7 describe the trend of our model(D-GAE) on the Ml-100k test loss and test RMSE
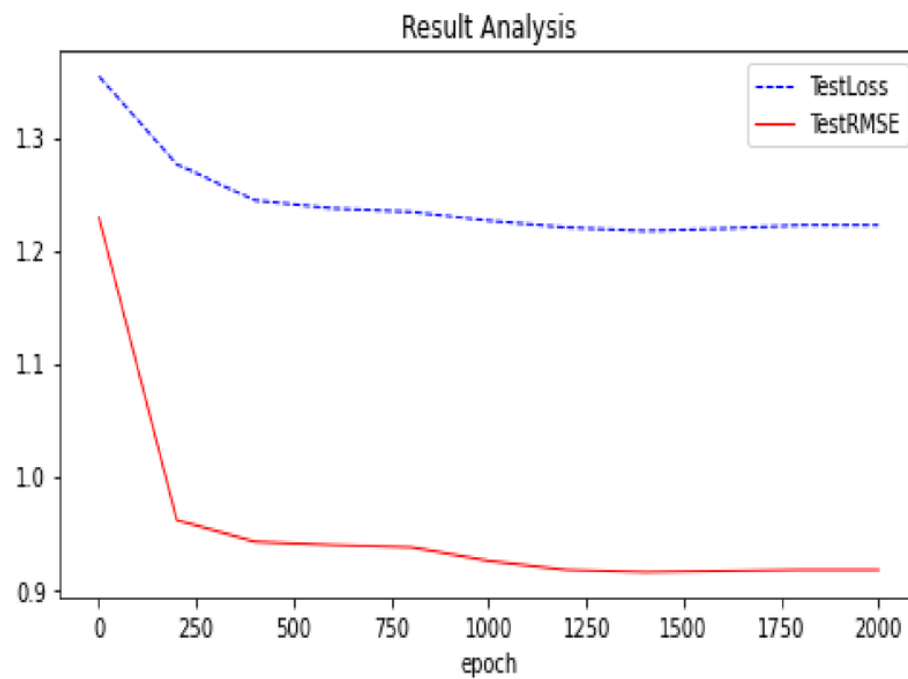
Figure 7. Experimental results (D-GAE basis on Ml-100k).

Figure 8

Experimental results on three datasets

Figure 8 describe the comparison of our model with other models on the other three

datasets.

Model	Flixster	Douban	YahooMusic
GRALS	1.313/1.245	0.833	38.0
sRGCNN	1.179/0.926	0.801	22.4
GC-MC	0.941/0.917	0.734	20.5
IGC-MC	0.909	0.990	21.3
PinSage	0.954	0.739	22.9
F-EAE	0.908	0.739	20.0
GAE+L1(ours)	0.926/0.911	0.730	19.2
GAE+L2(ours)	0.924/0.911	0.729	19.3
GAE+L1+L2(ours)	0.930/0.907	0.728	19.1
D-GAE(ours)	0.934/0.917	0.731	19.3

Figure 9

Noise contrast ablation experiment

Figure 9 depicts a comparative ablation experiment with the addition of different proportions of noise

	10%	20%	30%
Douban	0.734	0.731	0.735
Flixster	0.933/0.921	0.934/0.917	0.937/0.909
YahooMusic	19.6	19.3	20.0

Figure 10 Noise contrast ablation experiment