Deep Graph Embedding for Ranking Optimization in E-commerce

Chen Chu*
Alibaba Group
Hangzhou, China
chuchen.cc@alibaba-inc.com

Zhao Li†
Alibaba Group
Hangzhou, China
lizhao.lz@alibaba-inc.com

Beibei Xin‡
Departments of Biological Sciences and Computer Science, University of Southern California
Los Angeles, United States
bxin@usc.edu

Fengchao Peng§
Department of Computer Science and Engineering, Hong Kong University of Science and Technology
Hong Kong, China
fpengaa@cse.ust.hk

Chuanren Liu
Decision Sciences and MIS, Drexel University
Philadelphia, United States
chuanren.liu@drexel.edu

Remo Rohs
Departments of Biological Sciences and Computer Science, University of Southern California
Los Angeles, United States
rohs@usc.edu

Qiong Luo
Department of Computer Science and Engineering, Hong Kong University of Science and Technology
Hong Kong, China
luo@cse.ust.hk

Jingren Zhou¶
Alibaba Group
Hangzhou, China
jingren.zhou@alibaba-inc.com

ABSTRACT

Matching buyers with most suitable sellers providing relevant items (e.g., products) is essential for e-commerce platforms to guarantee customer experience. This matching process is usually achieved through modeling inter-group (buyer-seller) proximity by e-commerce ranking systems. However, current ranking systems often match buyers with sellers of various qualities, and the mismatch is detrimental to not only buyers' level of satisfaction but also the platforms' return on investment (ROI). In this paper, we address this problem by incorporating intra-group structural information (e.g., buyer-buyer proximity implied by buyer attributes) into the ranking systems. Specifically, we propose **De**ep **Gr**aph **E**mb**e**dding (DEGREE), a deep learning based method, to exploit both inter-group and intra-group proximities jointly for structural learning. With a sparse filtering technique, DEGREE can significantly improve the matching performance with computation resources less than that of alternative deep learning based methods. Experimental results demonstrate that DEGREE outperforms state-of-the-art graph embedding methods on real-world e-commence datasets. In particular, our solution boosts the average unit price in purchases during an online A/B test by up to 11.93%, leading to better operational efficiency and shopping experience.

CCS CONCEPTS

• **Computing methodologies** → **Learning latent representations**; **Neural networks**;

KEYWORDS

Deep Learning; Graph Embedding; Structure Learning; E-commerce Ranking; Customer Matching; A/B Test

ACM Reference Format:
Chen Chu, Zhao Li, Beibei Xin, Fengchao Peng, Chuanren Liu, Remo Rohs, Qiong Luo, and Jingren Zhou. 2018. Deep Graph Embedding for Ranking Optimization in E-commerce. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3269206.3272028

1 INTRODUCTION

Providing efficient shopping experience with quality items (e.g., products) is an important goal of e-commerce platforms [28]. This task is achieved through the design of e-commerce ranking system to match buyers with items based on relevance of items as well as preference of buyers. For example, as shown in Figure 1, a ranking

*first author.
†Corresponse and Co-first author.
‡Co-first author.
§Co-first author.
¶Corresponse author.

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3272028

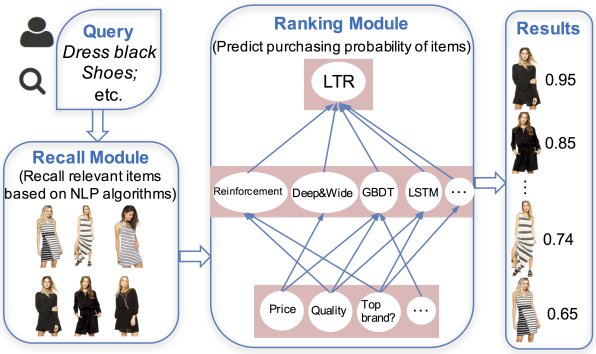

Figure 1: A general workflow of current ranking system. It consists of a query module to allow buyers to search items of interest, a recall module to identify items relevant to the query, and a ranking module to predict click/purchase probabilities of the identified items.

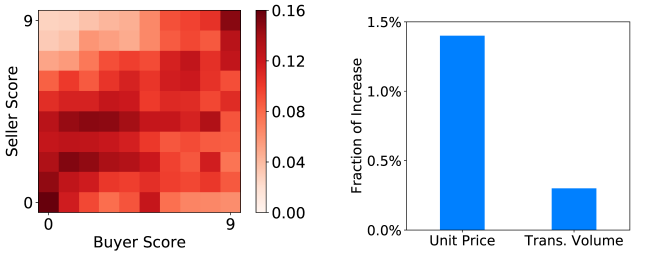

Figure 2: (a) The heat map displays transaction volumes between buyers and sellers across score levels (from low to high). The transaction volumes in the upper left and bottom right areas are considered less suitable matches. (b) The increase of the average unit price and the transaction volume in an online A/B test in which items from low-score sellers are removed in the ranking results for high-score buyers.

system first forwards the buyer's query to a recall module, which can identify relevant items from large database by understanding user's intentions with nature language processing (NLP) algorithms. Then, a ranking module will predict click/purchase probabilities for the relevant items with algorithms such as Wide & Deep learning [10], Reinforcement Learning [23], Gradient Boosted Decision Tree (GBDT) [15], and Long ShortTerm Memory (LSTM) [32]. Finally, the ranking system will display the relevant items to the buyer with descending click/purchase probabilities of the items. Following this general workflow, the ranking system can improve shopping efficiency on e-commerce platforms.

However, current e-commerce ranking system might match a buyer with items and sellers of various qualities. To investigate such issues, we quantify the quality of buyers and sellers with *buyer scores* and *seller scores*, respectively. These scores are predicted by supervised machine learning models with training data labeled by human experts. Generally speaking, a buyer (or seller) is labeled and predicted with high quality score if the buyer (or seller) often clicks/purchases (or sells) items with high price or items from luxury brands. With these scores, we examine the distribution of transaction volume between buyers and sellers of different qualities. As shown in Figure 2(a), we can tell that our current ranking system often match a buyer with sellers of various quality. Nonetheless, high-score buyers expecting high quality products and services may not want to be exposed to or purchase items from low-score sellers. Meanwhile, it might not be a pleasant experience if low-score buyers are matched with items containing better yet unaffordable items that render more suitable items less attractive. In both cases, the mismatches between buyers and sellers could result in sub-optimal shopping experiences as well as loss of sale opportunities.

Intuitively, shopping experience can be improved by reducing these less suitable matches. To validate this intuition we conducted an online A/B test for high-score buyers. In the A/B test, buyers are divided into two groups, i.e. a control group and a experiment group. In the control group, the matches are generated by the current ranking system while in the experiment group items from low-score sellers are removed from the matching results. Figure 2(b) shows that, by reducing less-suitable matches, both the transaction volume and the average unit price would increase with matched items tailored to the taste of buyers. We also observed that, this strategy enhances coherence in the matching results: similarly-scored buyers (or sellers) have close proximity in their profiles, and should be matched with sellers (or buyers) who also have close proximity in profiles. By exploiting dominant interactions between buyers and sellers and enhancing such coherence in the matching results, we could systematically address the mismatching problems in the current e-commerce ranking system.

Therefore, we exploit proximity of buyers (and sellers) as intra-group structural information for better e-commerce matching to ensure quality shopping experience. In contrast, most current ranking systems focus on exploiting inter-group network structure of buyers and sellers from explicit *buyer-seller* proximity determined by transaction records. The general ideal of network structural learning is the so-called graph embedding, such as matrix factorization [3], SDNE [30], network reconstruction [21, 30], and graph clustering [11, 27, 31, 33]. Each graph embedding method maps nodes in a network to low-dimensional embedding vectors by preserving certain network properties. There are three main categories of graph embedding approaches, namely factorization methods, random walks, and deep learning based methods. Factorization methods, such as locally linear embedding [24], graph factorization [3], Laplacian Eigenmaps [5], GraRep [8], and HOPE [21], maintain connection strengths in a network. Random walk approaches preserve community structure and allow computational parallelization. DeepWalk [22], node2vec [13] and APP [34] are representatives of this category. Deep learning based methods, such as SDNE [30] and DNGR [9], have recently attracted increasing attention by capturing non-linearity in networks. According to a recent survey [12], deep learning based methods outperform methods in other categories in many tasks, such as network clustering and visualization.

To address the mismatching problems, we design an effective deep learning framework by incorporating intra-group (*buyer-buyer* and *seller-seller*) information and learning non-linear network structures. The embedding produced by our framework tends to preserve: 1) intra-group network structure of sellers (buyers) based on seller-seller (buyer-buyer) proximity implied by their attributes; 2) inter-group network structure based on a pair-wise matrix describing

explicit interactions between buyers and sellers (e.g. transactions). The resulting embedding algorithm, **De**ep **Gr**aph **E**mb**e**dding or DEGREE, has two main advantages in comparison with current graph embedding methods. First, DEGREE obtains embedding vectors by learning both inter- and intra-group structures. Second, among deep learning based methods, DEGREE is scalable in both memory and running time with a sparse filtering technique. The sparse filtering uses sparse input for each vertex rather than rows of the interaction matrix, and learns fewer parameters due to a simpler network architecture.

We have conducted experiments on real-world e-commerce datasets. The results indicate that DEGREE compares favorably with other graph embedding methods in terms of learning accuracy as well as ability to identify dominant interactions between buyers and sellers and eventually reduce less suitable matches. In addition, DEGREE scales much better than existing deep learning based methods for graph embedding. Finally, we have evaluated DEGREE in a real e-commerce market through large-scale online A/B test. The results show that DEGREE significantly increases the average unit price in purchases during the A/B test. This increase will lead to not only improved shopping experience of our customers but also better operational efficiency of our e-commerce platform. All these strong empirical results clear validate our research motivation in this study, and show that by considering implicit intra-group structures, our DEGREE algorithm can improve current ranking system for large-scale e-commerce services.

In summary, our contributions in this work are as follows:

- We propose DEGREE, a deep learning based method, to perform graph embedding of buyers and sellers in e-commerce network. Our method is effective not only in identifying dominant inter-group interactions but also in providing suitable e-commerce matches by preserving both inter-group and intra-group network structures.
- We develop a sparse filtering technique and use one-hot vertex representation as sparse input to improve scalability of the deep learning based graph embedding. As a result, DEGREE outperforms other deep learning based methods on both memory and running time.
- We have evaluated DEGREE in real-world e-commerce datasets and large-scale online A/B tests. According to the results, our method outperforms current ranking system by producing more tailed buyer-seller matches which significantly increase the average unit price in purchases and improve shopping experience.

The remainder of our paper is organized as follows. In Section 2 we present basic definitions and a review of related work. In Section 3 we introduce our method in detail. The experimental results are shown in Sections 4 and the online A/B test results are shown in 5. Finally, we conclude this paper in Section 6.

2 BACKGROUND AND RELATED WORK

In this section, we first define the graph embedding problem. Then we introduce existing non-deep learning based methods as well as deep learning based methods.

Symbols	Definition
M, N	Number of buyers, sellers
k	Embedding dimension
D	Depth of DNN
R	Interaction matrix
G_B, G_S	Buyer/seller proximity matrix
X_B, X_S	One-hot encoding matrix of B, S
H_B^d, H_S^d	Output of d-th hidden layer
$W_B^d, W_S^d, \epsilon_B^d, \epsilon_S^d$	The d-th layer weight and bias
B, S	Embedding for buyers, sellers

Table 1: Terms and notations.

2.1 Definitions

DEFINITION 1. E-commerce network. Given a group of buyers V_B, and a group of sellers V_S from an e-commerce platform, where $V_B = \{V_{B1}, V_{B2}, \cdots, V_{BM}\}$ and $V_S = \{V_{S1}, V_{S2}, \cdots, V_{SN}\}$, an e-commerce network with V_B and V_S is defined as a graph $G = (V, E)$, a bipartite network on V_B and V_S, with nodes $V = V_B \cup V_S$ and edges $E = \{(V_{Bi}, V_{Sj})\}$ where $V_{Bi} \in V_B$ and $V_{Sj} \in V_S$.

In this e-commerce network, V_B and V_S are buyer and seller scores describing their attributes. Based on these scores, buyer-buyer proximity and seller-seller proximity can be derived, described as G_B and G_S, respectively. These proximities provide information to study intra-group structure of buyers and sellers. Moreover, connection strength describing historical transaction volume between V_B and V_S is denoted in a matrix R.

DEFINITION 2. Network Embedding. Given a bipartite network denoted as $G = (V_B \cup V_S, E)$, the network embedding consists of a mapping function

$$f_B : V_{Bi} \rightarrow B_i \in \mathbb{R}^k$$

and a function

$$f_S : V_{Sj} \rightarrow S_j \in \mathbb{R}^k,$$

where $k \ll max(|B|, |S|)$. Objectives of f_B and f_S are to preserve the (i) inter-group structure between V_B and V_S denoted in matrix R; (ii) implicit intra-group structure of V_B through G_B; (iii) implicit intra-group structure of V_S through G_S.

2.2 Non-deep Learning Based Approaches

In this subsection, we introduce non-deep learning based approaches in three categories, i.e. factorization based, random walk based, and others not categorized. Factorization based methods represented pioneering work for network embedding. Both locally linear embedding [24] and graph factorization [3] maintain connection strengths in an adjacency matrix, whereas Laplacian Eigenmaps [5] approximate a Laplacian matrix. Random walk approaches, such as DeepWalk [22], node2vec [13], and APP [34], capture higher-order proximities and are especially useful in node prediction in partially visible networks. DeepWalk preserves community information by defining sequences of short random walks for each node. Its local exploration of networks allows computational parallelization and high accommodation of dynamic networks. node2vec, however, focuses more on flexible exploration of neighbors to learn richer representations. APP is short for asymmetric proximity preserving and is a random walk based method that can preserve asymmetric similarities between vertex pairs. LINE [29], is a not categorized

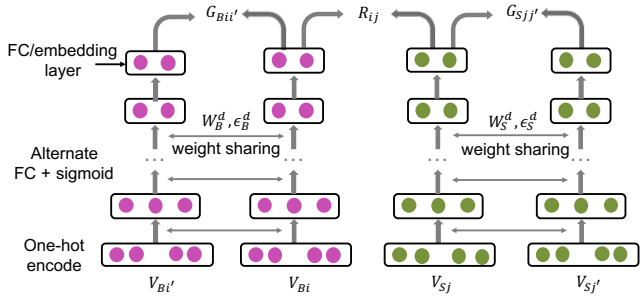

Figure 3: Deep neural network architecture of DEGREE. The leftmost block (magenta) learns buyer-buyer proximity in G_B, the rightmost block (green) learns seller-seller proximity in G_S, and the block in the center learns buyer-seller proximity in R. "FC" stands for fully-connected layer. The activation function used in DNN is a sigmoid function.

method, defines joint probability distributions for attributes (i.e. embeddings and interactions in adjacency matrix) for each pair of nodes, then minimizes Kullback-Leibler divergence of these distributions. Note that for e-commerce networks, these methods only consider inter-group structure.

2.3 Deep Learning Based Approaches

Deep neural networks have outperformed others graph embedding methods [12] due to their ability to model non-linear relationships in large-scale data. For example, DNGR [9] first derives a pointwise mutual information (PMI) matrix [18], a dense matrix reflecting pairwise interactions of nodes. It then utilizes a deep auto-encoder, which is an unsupervised learning framework to learn low dimensional representations [4]. SDNE [30] further uses a semi-supervised framework to capture first- and second-order proximities in a network. The unsupervised component uses a deep auto-encoder to learn embeddings, aiming at reconstructing neighbors for each node. The supervised component applies penalties when similar vertices are embedded far from each other. Their experiments showed that DNN-based embedding methods outperformed non-deep methods in various tasks [6, 7, 19]. Note that DNGR is computationally expensive in the first step and its second step is the same as SDNE, i.e. taking rows of the adjacency matrix as input. When the number of vertices becomes large, these techniques face scalability problems. Most importantly, the implicit intra-group structure of each group involved in e-commerce bipartite networks is rarely learnt and considered.

3 DEEP GRAPH EMBEDDING

In this section, we propose a DNN model, DEGREE, to perform e-commerce bipartite network embedding. Fig. 3 illustrates our DNN model containing two kinds of blocks, one for capturing intra-group relationship within buyer objects (magenta) and the other for seller objects (green). To preserve inter-group structure between sellers and buyers, we build connections between these two kinds of blocks as classical network embedding techniques. A difference is that DEGREE takes one-hot encoding of vertex IDs as input. Through alternating between several fully connected layers and activation layers in each block, as well as a fully connected layer at the top, it learns valuable embedding vectors in the latent space at the last layer. During training, we conducted weight sharing within buyer seller blocks. By jointly optimizing the captured relationships in both blocks and the proximity between buyers and sellers, DEGREE can better learn bipartite network representations and is scalable to large networks. In the following, we introduce the implementation details of DEGREE with specific focus on how DEGREE solves the matching problem in e-commerce.

3.1 Inter-group Structure

Our first goal is to ensure that the embedding capture well the interactions between buyers and sellers. We adapt the idea of MF to build the model. We assign one embedding vector B_i for each buyer vertex V_{Bi}, and one embedding vector S_j for each seller vertex V_{Sj}. Then, given the interaction matrix $R \in \mathbb{R}^{M \times N}$, MF finds the matrices $B \in \mathbb{R}^{M \times k}$ and $S \in \mathbb{R}^{N \times k}$ ($k \ll min(M, N)$) such that

$$R \approx B \cdot S^T \tag{1}$$

However, MF can only find linear embeddings for vertices. In order to exploit non-linear embeddings, instead of directly applying typical MF, we propose to first perform non-linear embedding to the vertices through a DNN method, and then use the embedding results to approximate the interaction matrix. In particular, we build a multi-layer fully connected network with sigmoid activations[14]. We assign a one-hot encoding for each vertex. And then we use the one-hot encoding as input of DNN. For buyer vertex V_{Bi} and seller vertex V_{Sj}, the representation of each hidden layer is as follows:

$$\begin{aligned} h_{Bi}^{(1)} &= \sigma(W_B^{(1)} x_{Bi} + \epsilon_B^{(1)}) \\ h_{Bi}^{(d)} &= \sigma(W_B^{(d)} h_{Bi}^{(d-1)} + \epsilon_B^{(d)}) \\ h_{Sj}^{(1)} &= \sigma(W_S^{(1)} x_{Sj} + \epsilon_S^{(1)}) \\ h_{Sj}^{(d)} &= \sigma(W_S^{(d)} h_{Sj}^{(d-1)} + \epsilon_S^{(d)}) \end{aligned} \tag{2}$$

where x_{Bi} and x_{Sj} are the one-hot encoding of vertices V_{Bi} and V_{Sj}. $\sigma(\cdot)$ stands for the sigmoid function.

The representation of the last layer is defined as follows:

$$\begin{aligned} B_i &= W_B^{(D)} h_{Bi}^{(D-1)} + \epsilon_B^{(D)} \\ S_j &= W_S^{(D)} h_{Sj}^{(D-1)} + \epsilon_S^{(D)} \end{aligned} \tag{3}$$

The difference between the last layer and other layers is that we do not use an activation function in the last layer, such that the output of the last layer can be of any scale, rather than between 0 and 1, to approximate R. We use the output of the last layer as the non-linear embedding of each vertex. Then we define the loss function of learning the inter-group structure as follows:

$$L_R = ||R - B \cdot S^T||_F^2 \tag{4}$$

3.2 Intra-group Structure

After discussing inter-group structural learning, we incorporate additional intra-group information, buyer-buyer proximity in G_B and seller-seller proximity G_S, into network embedding. In our target platform, each element $G_B(i, j)$ is the absolute difference

between buyer scores of buyers V_{Bi} and V_{Bj}. Similarly, we build the pairwise seller-seller proximity matrix G_S for sellers.

We use the matrices G_B and G_S in the model and propose the following loss function for intra-group structure:

$$L_{BS} = \lambda_B L_B + \lambda_S L_S, \tag{5}$$

where

$$L_B = ||BB^T - G_B||_F^2,$$

and

$$L_S = ||SS^T - G_S||_F^2.$$

The term L_B (L_S) aims to guarantee that the representation vectors of buyers (sellers) in the latent space can preserve the original intra-group structure. We integrate these intra-group regularization terms for better structural learning in an e-commerce network. Our experiments and online A/B tests will show that DEGREE reduces the less suitable matches between buyers and sellers and improves shopping experience on the e-commerce platform.

3.3 Sparse Filtering

In the loss function *eq.*(5), both matrices G_B and G_S are dense. Their sizes are quadratic to the number of vertices in the graph, which significantly decreases the scalability of our method. In order to improve the scalability and reduce the storage cost, we propose a sparse filter on the dense matrices G_B and G_S such that after the filtering, we only need to calculate the values of two sparse matrices. The time cost is reduced to near-linear time.

For G_B, instead of calculating the proximity between every pair of buyers, we only consider the proximity between pairs of buyers that have bought items from at least one common seller. This strategy can be achieved by finding each buyer's neighbor set in the e-commerce network.

For each buyer, we call the set of buyers that have common neighbors with this buyer as the 2-hop neighbor set. With this set, we only need to calculate proximities between this buyer and each 2-hop neighbor, rather than all the buyers. This method effectively reduces computation, however, one problem is that the number of neighbors of each buyer can vary from tens to thousands. The imbalance deteriorates in the 2-hop neighbors because a buyer with more neighbors will have an even larger number of 2-hop neighbors. To alleviate the data imbalance and to further reduce the data size, we sampled from the 2-hop neighbor set of each buyer. In particular, in each 2-hop neighbor set, we first sorted the buyers based on the Adamic/Adar Similarity (Adar) [2], which is defined as follows:

$$Adar(u, v) = \sum_{w \in \Gamma(u) \cap \Gamma(v)} \frac{1}{\log |\Gamma(w)|} \tag{6}$$

where $\Gamma(u)$ is the neighbor set of vertex u in the graph. The Adar similarity is widely applied to calculate the similarity between vertices in a graph. It yields high similarity to two vertices that share some rare neighbors, and is more effective than similarity metrics that only consider the count of common neighbors. We select the set of buyers with top K Adar values, and only calculate the proximity between each buyer and each of the 2-hop neighbors with top K Adar values.

We perform a similar sampling strategy on G_S. Then we define two sparse matrices Q_B and Q_S such that $Q_B(i, j) = 1$ if the buyer pair (V_{Bi}, V_{Bj}) is selected by our sampling process, and $Q_B(i, j) = 0$ otherwise. These two matrices work as sparse filters to the dense matrices G_B and G_S, respectively. Given Q_B, Q_S, the loss function of intra-group structure is re-defined as follows:

$$L_{BS} = \lambda_B ||Q_B \odot (BB^T - G_B)||_F^2 + \lambda_S ||Q_S \odot (SS^T - G_S)||_F^2 \tag{7}$$

where $\odot$ is the Hadamard product.

The sparse filtering slightly increases the computational overhead, but it significantly reduces space cost of our method. Taking a buyer vertex as an example, we present the cost analysis in the following. The time cost of finding neighbors for each V_{Bi} is $O(M|R|)$. The time cost of finding 2-hop neighbor sets is $O(M \cdot N_{max}^2)$, where N_{max} is the maximum number of neighbors of any vertex in the graph. We only calculate the proximity between the buyer-seller pairs that are selected by the sparse filter, and the time cost is $O(MK)$. Since the rating graph is highly sparse, we have $N_{max} \ll min\{M, N\}$. Therefore, the dominating time cost for computing proximity with sparse filtering is $O(M|R|)$. In comparison, the time cost for computing pairwise proximity without sparse filtering is $O(M^2)$. We find that the time complexity of sparse filtering is higher than that of directly computing the pairwise proximity because $|R| > M$. However, without sparse filtering, the entire proximity matrix would be used in calculating the loss in each iteration, and the time complexity is $O(M^2)$. With sparse filtering, the time complexity is merely $O(MK)$ because only the sampled proximity values are used. Moreover, sparse filtering significantly reduces the space cost because it only needs to store the sparse proximity matrix, instead of the entire dense matrix. In practice, we find that the computational overhead of sparse filtering is compensated by the time cost saved during the optimization process because of the sparse matrix. The memory cost is also significantly reduced.

3.4 Scalability

Scalability is a critical problem in large scale graph embedding [20, 25, 26]. In the following, we use three methods to improve the scalability of our approach. First, we convert the input of the model to a sparse format. For input $V_{Bi'}, V_{Bi}, V_{Sj}, V_{Sj'}$, we use one-hot encoding of the vertex IDs and stored each batch of input in the sparse format. In this way, we eliminate large adjacency matrices, or dense PPMI matrices, which are used in previous work such as SDNE and DNGR. When the number of buyers/sellers increases, the memory cost increases quadratic in SDNE and DNGR, but the memory cost of DEGREE only increases linearly. Second, supervised learning of R, G_B, and G_S in DEGREE avoids the need to store large dense matrices as intermediate results in DNN. In contrast, deep auto-encoders used in SDNE and DNGR output dense matrices at the output layer. As such, our DEGREE's memory consumption is orders of magnitude lower than both SDNE's and DNGR's. Third, to obtain embedding vectors of the same dimension, fully-connected neural networks in our DNN model have a simpler network architecture than deep auto-encoders in unsupervised learning. Fewer layers and weights further largely speed up the training process.

3.5 Optimization

We add regularization on the embedding vectors and model parameters to prevent overfitting:

$$L_{reg1} = \frac{1}{2}(\|W_B\|_F^2 + \|W_S\|_F^2),$$
$$L_{reg2} = \frac{1}{2}(\|B\|_F^2 + \|S\|_F^2) \tag{8}$$

The complete loss function is defined as follows:

$$L = L_R + L_{BS} + \alpha_1 L_{reg1} + \alpha_2 L_{reg2} \tag{9}$$

The components L_R, L_{BS}, L_{reg1} and L_{reg2} are calculated respectively in $eq.(4)$, $eq.(7)$ and $eq.(8)$, respectively. We choose the adaptive gradient descent method Adam [16] as the optimization algorithm.

4 EXPERIMENTS

We conduct experiments on three real world e-commerce datasets. Our proposed model DEGREE is compared with several existing methods. All experiments are executed on a 2.30GHz Intel(R) Xeon(R) CPU with 64 GB memory, running on Fedora 7.2. All methods are implemented using Python with Tensorflow [1].

4.1 Datasets

The three datasets are provided by a real world e-commerce platform. They are collected from three item categories, namely Women Apparel, Women Shoes, and Digital Products. Each dataset contains all the transactions of 100,000 buyers that are randomly selected from the pool of all buyers. Each data record consists of a buyer ID V_{Bi}, a seller ID V_{Si} and the transaction volume between them R_{ij}. We also have scores for each seller and buyer in the datasets which are used to model intra-group structure in our method. Not all pairs of buyers and sellers are considered in modeling. Only those pairs determined by sparse filtering in Section 3.3 are used, resulting in sparse G_B and G_S. The basic information of the dataset is shown in Table 2. Note that missing values in G_B, G_S, and R are not modeled. Different graph embedding methods have different format of training data. Specifically, each sample of input data for DEGREE is a four-element tuple ($V_{Bi'}$, V_{Bi}, V_{Sj}, $V_{Sj'}$), with a matched three-element output ($G_{Bii'}$, R_{ij}, $G_{Sjj'}$) (Fig. 3).

4.2 Existing Methods

We compared DEGREE with the following existing methods:

1. **MF**. MF is frequently used in recommendation systems [3, 17]. Every piece of input data for MF is a two-element tuple (V_{Bi}, V_{Sj}), with a matched R_{ij} as output.

2. **Graph Regularized Matrix Factorization (GRMF)**. GRMF is an extension of MF that considers both inter-group structure and intra-group structure. GRMF has similar input and output data with DEGREE.

3. **SDNE**. SDNE [30] learns graph embeddings which preserves the first-order and second-order proximity. It is powerful in graph representation through introducing auto-encoders to model second-order proximities. However, it only considers the inter-group structure as MF does. SDNE performs supervised and unsupervised learning on rows of an adjacency matrix which is usually large and sparse in e-commerce networks. Instead of directly storing an $M \times N$ adjacency matrix, we implemented SDNE-sparse, and feed the sparse format of adjacency rows to SDNE auto-encoders. In the following, we will use SDNE and SDNE-sparse interchangeably. SDNE has similar input and output data with MF.

4. **DNGR**. DNGR [9] is a graph embedding method based on deep neural networks. It first computes a PPMI matrix based on random surfing. The PPMI matrix is then fed to a stacked denoising auto-encoder to learn the node embeddings. It suffers from the problem of high computation complexity and memory cost either in calculating the PPMI matrix or in modeling the auto-encoder.

	# of Buyers	# of Sellers	# of Records
Women Apparel	100000	85856	583365
Women Shoes	100000	45952	579224
Digital Products	100000	40780	536038

Table 2: Basic information of datasets.

Parameter Setting: The parameters in all of the methods are manually tuned to the best performance. The embedding size of all the methods is set to 16. In MF, the learning rate is set to 0.05. In GRMF, the learning rate is also 0.05, and the parameter controlling the intra-group regularization is set to 1.0. In SDNE, we use three layers, each containing 64, 32, and 16 neurons respectively. The other parameters of SDNE are the same as those in the original work. DEGREE also uses 3 layers, each with 64, 32, and 16 neurons. The parameters α_1 and α_2 are both set to 10^{-5}. For simplicity, λ_B is set to be equal to λ_S. We do not perform DNGR experiments because of the problem of memory cost and computation complexity. However, we compare its memory cost in the experimental results.

4.3 Metrics for Performance Evaluations

RMSE (root mean square error) is introduced to measure the effectiveness of maintaining explicit inter-group structure between buyers and sellers in the network. To further evaluate the capability of uncovering dominant interactions, we compare the proportions of less suitable matches derived by all methods, i.e. the areas of top left and bottom right in Fig. 2 (a).

Buyers and sellers are matched by the K-nearest neighbors in the network, i.e. we keep K-nearest buyers for each seller and K-nearest sellers for each buyer. The K is set to 50 in the experiments. To calculate the proportion of unsuitable matches, we first split buyers and sellers into ten ordered bins based on their scores. For each pair of buyer bin i and seller bin j, we calculate the pair's transaction volume $T(i, j)$ as the sum of all the transaction volumes generated by each buyer and his K-nearest sellers.

Based on the transaction volume T, we define the target proportion as Less Suitable Match Ratio ($LSMR$) as follows:

$$LSMR = \frac{\sum_{\|i-j\|>=\delta} T(i,j)}{\sum_{i,j=1}^{10} T(i,j)} \tag{10}$$

4.4 Performance

The RMSE results are presented in Table 3. DEGREE achieves the lowest RMSE on all the datasets whereas GRMF performed better than MF. The reason is that the rich information embedded in buyer

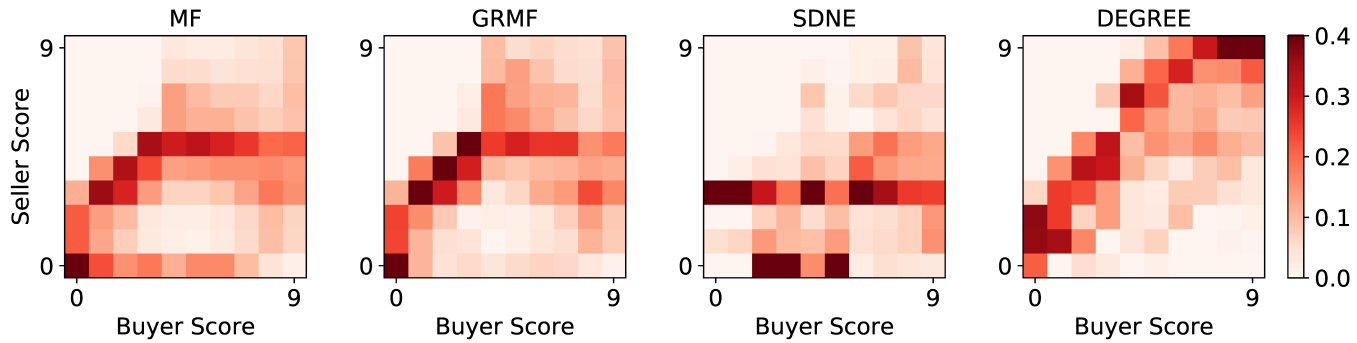

Figure 4: The heat map of transaction volumes between matched buyers and sellers. The result of DEGREE shows peak volumes along the diagonal whereas the other three methods do not, indicating that DEGREE better boosts the transaction volumes between buyers and sellers of similar score levels in the latent space.

	MF	GRMF	SDNE	DEGREE
Women Apparel	1.612	1.314	N/A	**0.995**
Women Shoes	1.162	1.005	N/A	**0.970**
Digital Products	0.884	0.831	N/A	**0.824**

Table 3: The RMSE of each method on each dataset.

	MF	GRMF	SDNE	DEGREE
Women Apparel	0.178	0.176	0.250	**0.145**
Women Shoes	0.272	0.112	0.209	**0.066**
Digital Products	0.224	0.212	0.372	**0.121**

Table 4: LSMR of each method on each dataset. A lower value indicates a better result on reducing less suitable matches.

scores and seller scores can be useful to uncover the explicit interactions between buyers and sellers in the network. Since DEGREE and GRMF both exploit the inter-group and intra-group structures, they outperform MF. Moreover, DEGREE is able to capture the non-linearity in the buyer-seller network, it performs even better than GRMF.

Note that RMSE of SDNE is not presented in Table 3. The reason is that the objective of SDNE is different from other methods. SDNE considers the explicit interactions, i.e. transaction volumes in experimental datasets, as the weights of edges in the buyer-seller network. It is not designed to predict the edge weight.

We also draw heat maps of $T(i, j)$ for buyer bin i and seller bin j for each method in Fig. 4. The results are obtained on the Women Apparel dataset. The heat map generated by DEGREE shows a clear diagonal characteristic whereas those of the other methods do not. The results indicate that DEGREE performs best in reducing less suitable matches and in uncovering dominant interactions between sellers and buyers. As clearly visible in Table 4, DEGREE outperforms other methods *LSMR* with δ setting to five. Moreover, GRMF outperforms MF and SDNE because similar to DEGREE, GRMF considers inter-group and intra-group structure. DEGREE outperforms GRMF in that it uses a DNN-based model to exploit non-linear structure, whereas GRMF can only learn linear structure. In the interest of space, we do not present *LSMR* with all possible δ values though DEGREE still achieves the smallest *LSMR* with other δ settings.

λ_B, λ_S	0.01	0.1	1.0
Women Apparel	0.161	0.153	0.145
Women Shoes	0.049	0.051	0.066
Digital Products	0.129	0.112	0.121

Table 5: LSMR of DEGREE with different parameters.

Memory				
MF	GRMF	DEGREE	SDNE	DNGR
130M	250M	320M	4G	>100G
Running time/batch				
Batch size	MF	GRMF	DEGREE	SDNE
10000	0.047s	0.064s	0.099s	N/A
3000	0.015s	0.021s	0.077s	20.27s
500	0.0031s	0.0049s	0.066s	3.03s

Table 6: Comparison of memory usage and running time.

4.5 Parameter Sensitivity

The key parameters of our model are λ_B and λ_S. In the experiments, we simply set the two parameters to be equal. We compare the diagonal ratio with different parameter settings in Table 5. We find that the optimal parameter settings for different datasets may vary. The *LSMR* values of DEGREE alter slightly and are always lower than those of the other methods, as shown in Table 4.

4.6 Memory Consumption and Running Time

To illustrate the scalability of DEGREE, in Table 6 we compare the memory consumption and running time of each method based on the Women Apparel dataset. The methods MF, GRMF, SDNE, and DEGREE all consider an adjacency matrix in their loss functions. However, our experiment dataset contains more than 100,000 vertices, which requires as much as over 90 GB of space for the corresponding adjacency matrix. To improve the scalability, we convert the adjacency matrix into a sparse format first and feed it to the model. The time and space cost shown in Table 6 is the cost required after the improvement. However, DNGR requires a dense PPMI matrix, so that its space cost cannot be reduced in this way.

We compare the memory consumption and running time of each method under different settings of batch size. We find that in Table 6, the memory and time cost generally increases with the

(a) DEGREE

(b) Current system

Figure 5: Comparison of ranking results on experiment group and control group. (a) Top items in experiment group with DEGREE. Average price is CNY 764.60 and median price is CNY 639.00. (b) Top items in control group. Average price is CNY 417.50 and median price is CNY 386.00.

model complexity. Both of the memory and time cost of DEGREE is higher than that of GRMF, and GRMF's cost is higher than MF's cost, since DEGREE is more complex in model structure than the other two methods. Despite these differences, the costs of these three methods are of the same order of magnitude. However, both memory and time costs of SDNE are orders of magnitude higher than that of DEGREE. The reason of the expensive memory cost is that SDNE takes the entire adjacency matrix of the graph as direct input and introduces auto-encoder loss in the loss function. Though the input adjacency matrix can be compressed into sparse format, the auto-encoder loss can not. In addition to the computation of auto-encoder loss, the neural network structure of auto-encoder used by SDNE is much more complex than that of DEGREE, which further increases running time cost. The memory consumption of DNGR, also presented in Table 6, is even greater, since DNGR uses a dense PPMI matrix as input. Additionally, the calculation of PPMI matrix is high in computational complexity and space complexity. For these reasons, we did not perform other experiments using DNGR. The running time of SDNE with batch size as 10000 is not presented because the memory cost in this case is too high to run on our machine. Therefore, though the time cost and space cost of DEGREE is higher than the simple linear models, DEGREE is much more efficient than the existing deep learning models. As the batch size increases from 500 to 10000, the time cost of DEGREE increases slowly, from 0.066 sec/batch to 0.099 sec/batch.

5 ONLINE A/B TEST

We have performed an online A/B test for two weeks to evaluate the effectiveness of DEGREE on a real-world e-commerce platform. For simplicity, the test is focused on high-score buyers. The ranking results for queries of the control group in the test are provided by the original ranking system, which are ranked according to ranking scores of items. The results of the experiment group contain more items from closely embedded sellers of the buyers, which is achieved by adding an positive constant c_0 to ranking scores when sellers are among the K-nearest sellers of the buyers in the embedding space. Here c_0 is determined by human experts and is set the same for both GRMF and DEGREE for fair comparison.

Metric	Meaning
Price	Mean unit price of items bought by high-score buyer
DAHSB	Number of daily active high-score buyer
ACHSB	Average clicks per high-score buyer
HSB	Number of high-score buyers who have clicked items
IPV	Number of clicks
Trans. V	Total transaction volumes of high-score buyers

Table 7: Metrics measured in online A/B test

	Price	DAHSB	ACHSB	HSB	IPV	Trans. V
GRMF	7.46	1.46	1.51	1.37	1.83	-1.13
DEGREE	11.93	2.27	2.78	2.29	3.78	-0.71

Table 8: Results of online A/B test of GRMF and DEGREE. The values are the increasing percentages of the experiment group over the control group.

The ranking results suggest that our approach can improve the shopping experience of buyers by ranking more items from suitably matched sellers at the top. As we can see in Fig. 5, compared with the control group, the median price of the top 50 items in the experiment group is much higher in the ranking results of query *Dresses*. In addition, the ranking results of the experiment group contain famous brands of women apparel at the top such as MO&Co, whereas the control group only presents dresses of brands that are not so well-known. The results in the experiment group are more suitable for quality shopping experience since high-score buyers tend to have high purchasing power, prefer sellers that are selling high quality products.

We show the metrics measured in online A/B test in Table 7 and the detailed results in Table 8. We find that both GRMF and DEGREE increase the average unit price, yet DEGREE outperforms GRMF by increasing average unit price by 11.93% and by sacrificing less transaction volume[1]. The result demonstrates that DEGREE can improve the shopping experience of high-score buyers by exposing more quality items without reducing revenue of the e-commerce platform. This leads to several desired effects. First, the operational efficiency of the e-commerce platform (e.g., logistic and transportation) are improved since the number of sold items decreases when average unit price increases and total transaction volume remains flat. Second, DEGREE achieves greater increase in terms of DAHSB (daily active HSB), ACHSB (average clicks of HSB), HSB (number of high-score buyers), and IPV (item page viewed). The greater increase means that even though exposed items are more expensive in the experiment group, high-score buyers are more likely to be active on the e-commerce platform and to view more items, signaling more loyalty to the platform. Overall, our approach can improve not only shopping experience of our customers but also operational efficiency of our platform.

[1]Increasing or decreasing by less that 1% is considered insignificant in online A/B test on our e-commerce platform

6 CONCLUSION

In this paper, we propose DEGREE, a DNN-based graph embedding method to learn effective embeddings for buyers and sellers in e-commerce network and alleviate the problem of less suitable matches in the current ranking system in e-commerce market. DEGREE introduces new factors in the graph embedding loss function and enables the embeddings to preserve not only the inter-group structure between buyers and sellers but also the underlying intra-group structure of buyers (or sellers). Inter-group structure is interpreted as interactive activities between buyers and sellers whereas the intra-group structure is incorporated in the loss function through scores of buyers and sellers provided by the e-commerce platform. The resulting embedding vectors play an important role in identifying dominant interactions and providing better matches, especially for inactive buyers who have rare activities in the e-commerce market. A sparse filtering technique is adopted to reduce computational complexity and space complexity. As a result, the memory and time cost of DEGREE is orders of magnitude lower than that of existing DNN-based graph embedding methods. Results of experiments on real e-commerce datasets show that DEGREE outperforms existing methods in terms of RMSE and the capability of reducing less suitable matches and uncovering dominant interactions between buyers and sellers. The results of an online A/B test indicate that dominant interactions obtained by DEGREE can boost current ranking system on improving shopping experience and increasing the loyalty of targeted buyers.

As future work, we consider extending DEGREE to apply on more buyers in e-commerce platforms, rather than only on high-score buyers. We also consider to learn embeddings for vertices in more complex networks in e-commerce, where the edges and the vertices may be multi-typed.

ACKNOWLEDGMENTS

This work was supported by a Research Enhancement Fellowship from the USC Graduate School (to B.X.), and the National Institutes of Health (grant R01GM106056 to R.R.).

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467* (2016).

[2] Lada A Adamic and Eytan Adar. 2003. Friends and neighbors on the web. *Social networks* 25, 3 (2003), 211–230.

[3] Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J Smola. 2013. Distributed large-scale natural graph factorization. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 37–48.

[4] Leonardo Badino, Claudia Canevari, Luciano Fadiga, and Giorgio Metta. 2014. An auto-encoder based approach to unsupervised learning of subword units. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 7634–7638.

[5] Mikhail Belkin and Partha Niyogi. 2002. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems*. 585–591.

[6] Mikhail Belkin and Partha Niyogi. 2003. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation* 15, 6 (2003), 1373–1396.

[7] John A Bullinaria and Joseph P Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior research methods* 39, 3 (2007), 510–526.

[8] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 891–900.

[9] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2016. Deep Neural Networks for Learning Graph Representations.. In *AAAI*. 1145–1152.

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

[11] C. H. Q. Ding, Xiaofeng He, Hongyuan Zha, Ming Gu, and H. D. Simon. 2001. A min-max cut algorithm for graph partitioning and data clustering. In *Proceedings 2001 IEEE International Conference on Data Mining*. 107–114. https://doi.org/10.1109/ICDM.2001.989507

[12] Palash Goyal and Emilio Ferrara. 2017. Graph Embedding Techniques, Applications, and Performance: A Survey. *arXiv preprint arXiv:1705.02801* (2017).

[13] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[15] Michael Jahrer, Andreas Töscher, and Robert Legenstein. 2010. Combining predictions for accurate recommender systems. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 693–702.

[16] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[17] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009).

[18] Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*. 2177–2185.

[19] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *journal of the Association for Information Science and Technology* 58, 7 (2007), 1019–1031.

[20] Daniel A Menasce. 2000. Scaling for e-business. In *Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 2000. Proceedings. 8th International Symposium on*. IEEE, 511–513.

[21] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric Transitivity Preserving Graph Embedding.. In *KDD*. 1105–1114.

[22] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[23] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. Introduction to recommender systems handbook. In *Recommender systems handbook*. Springer, 1–35.

[24] Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *science* 290, 5500 (2000), 2323–2326.

[25] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2000. Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2nd ACM conference on Electronic commerce*. ACM, 158–167.

[26] Badrul M Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2002. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the fifth international conference on computer and information technology*, Vol. 1.

[27] Jianbo Shi and Jitendra Malik. 2000. Normalized Cuts and Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 22, 8 (Aug. 2000), 888–905. https://doi.org/10.1109/34.868688

[28] David M Szymanski and Richard T Hise. 2000. E-satisfaction: an initial examination. *Journal of retailing* 76, 3 (2000), 309–322.

[29] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1067–1077.

[30] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1225–1234.

[31] Scott White and Padhraic Smyth. [n. d.]. *A Spectral Clustering Approach To Finding Communities in Graphs*. 274–285. https://doi.org/10.1137/1.9781611972757.25 arXiv:http://epubs.siam.org/doi/pdf/10.1137/1.9781611972757.25

[32] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. 2017. Recurrent recommender networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 495–503.

[33] Xiaowei Xu, Nurcan Yuruk, Zhidan Feng, and Thomas AJ Schweiger. 2007. Scan: a structural clustering algorithm for networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 824–833.

[34] Chang Zhou, Yuqiong Liu, Xiaofei Liu, Zhongyi Liu, and Jun Gao. 2017. Scalable Graph Embedding for Asymmetric Proximity.. In *AAAI*. 2942–2948.