Multi-Task Learning on Heterogeneous Graph Neural Network for Substitute Recommendation

Tianchen Zhou
zhou.2220@osu.edu
The Ohio State University

Michinari Momma
michi@amazon.com
Amazon.com

Chaosheng Dong
chaosd@amazon.com
Amazon.com

Fan Yang
fnam@amazon.com
Amazon.com

Chenghuan Guo
chenghg@amazon.com
Amazon.com

Jin Shang
imjshang@amazon.com
Amazon.com

Jia Liu
liu.1736@osu.edu
The Ohio State University

ABSTRACT

Substitute recommendation in e-commerce has attracted increasing attention in recent years, to help improve customer experience. In this work, we propose a multi-task graph learning framework that jointly learns from supervised and unsupervised objectives with heterogeneous graphs. Particularly, we propose a new contrastive method that extracts global information from both positive and negative neighbors. By feeding substitute signal data from different sources to learning tasks with different focuses, our model learns the representation of products that can be applied for substitute identification under different substitutable criteria. We conduct experiments on Amazon datasets, and the experiment results demonstrate that our method outperforms all existing baselines in terms of comprehensive performance among all metrics of interest.

CCS CONCEPTS

• **Applied computing** → **Online shopping**.

KEYWORDS

Substitute Product Identification, Graph Neural Networks

ACM Reference Format:
Tianchen Zhou, Michinari Momma, Chaosheng Dong, Fan Yang, Chenghuan Guo, Jin Shang, and Jia Liu. 2023. Multi-Task Learning on Heterogeneous Graph Neural Network for Substitute Recommendation. In *Proceedings of 19th ACM SIGKDD Workshop on Mining and Learning with Graphs (MLG @ KDD'23).* ACM, New York, NY, USA, 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

As an emerging field in recommendation system, substitute recommendation has attracted growing attention in recent years [2, 11, 15, 18, 19, 23, 27, 28], with applications such as recommending subsitutes when a product is out-of-stock. Typically, substitutable products are similar and compatible, and equally attractive to target customers [15, 19]. Most of the existing methods learn the substitutable relation by optimizing a single loss according to multiple sources of information; Some works try to learn it by the product static features combined with customer feedback, such as specifications, prices, title, images and brands combined with product ratings and reviews [2, 15], and others learn it by customer behavior data, such as search and click logs, impression and purchase logs, etc [11, 28]. However, in practice, a single loss may not properly measure the substitutability in all aspects; rather, multiple substitute criteria are often desired to be optimized. Hence, the lack of a fundamental understanding on how to jointly learn from different substitute signals motivates us to fill this gap in this paper.

Intuitively, it is highly desirable to learn a better product representation from multiple substitute signals, so that we can measure the substitutable level of a pair of products by simply measuring the distance of their representations. Graph neural networks (GNN) is a popular tool for representation learning, which has been commonly used to learn node representations [7, 9, 15, 20, 29]. By aggregating information from neighbors along edges iteratively, one can learn integrated/global patterns traversed through the graph structure. Thus, GNN is widely used for substitute identification [11, 28], and graph clustering [3, 20, 24] to learn local as well as global patterns.

In this work, we propose a multi-task objective that consists of both supervised and unsupervised graph learning with loss functions that are independently defined for each type of different data sources. Our contributions are summarized as follows:

- We propose a multi-task graph learning model, named Heterogeneous Graph Infomax (HGI), that learns product representations from different sources, either supervised or unsupervised, where each learning task has its own objective and purpose. All the learning tasks are jointly optimized by combining their model parameters, and by tuning the task weight parameter, the output product representations can be applied for substitute identification under different substitutable criteria. Experimental results validate the effectiveness of HGI where each task can be improved without degrading performance for others.
- We propose a new objective function for unsupervised graph learning that is based on neighborhood information and learns from both positive and negative signals extracted from the graph structure. The objective is to optimize a contrastive loss between positive and negative neighborhood representations summarized from the graph structure.

Section 2 discusses related work. Section 3 introduces proposed learning framework. Section 4 gives experimental results. Section 5 concludes this paper.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
MLG @ KDD'23, August, 2023, Long Beach, CA
© 2023 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

2 RELATED WORK

Substitute recommendation. The concept of substitute has been proposed decades ago [1, 19], and such substitute relation was recently integrated in recommendation systems [11, 15, 28]. In particular, [15] developed the first method that models and predicts substitutable products by combining product reviews and product features via topic modeling. Meanwhile, deep learning models are proposed to model the substitutable relationship between products [18, 23, 25–27, 30]. Inspired by [7], [28] proposed a heterogeneous graph neural network, which learns substitute information from knowledge graph that is constructed by customer behavior data such as co-views and co-purchases. Typically, customer behavior data is viewed as product relationship, and is leveraged to connect product nodes in the graph. Although such behavior connections are easily obtained, there are limitations such as incomplete/noisy links due to low counting and cold start problems as it takes time for new products to receive customer behaviors. To address the issue, [11] proposed a multi-task objective that learns from multiple customer behaviors such as product search, etc. Although the objective in [11] consists of loss functions designed for each data, all learning tasks are learning from customer behavior data, which still can be incomplete/noisy. In our work, we propose a novel unsupervised learning task that learns from graph structure, which helps to mitigate the issue by learning global structures.

For a more general recommendation to recommend products to users (i.e., not product relationship such as substitutes), the Graph Convolution Networks have been applied [10, 12]. Notably, [10] applies clustering to avoid negative impact from higher order neighbors, which is similar to our work in terms of learning global structure. However, our work aims at achieving it via contrastive learning in a *soft* manner, avoiding expensive clustering computation as well as parameter tuning of clustering.

Contrastive learning. Contrastive learning is a popular approach in unsupervised learning, where a model is trained based on contrastive representations. Many existing works leverage contrastive learning to learn representations of nodes in a graph [6, 8, 17, 21]. However, when obtaining negative samples for contrastive purposes in objective, most of the existing works choose to randomly sample some negative nodes from the graph or utilize some random transformations, which "creates" the negative samples. In our work, as we have both positive and negative edges, we can leverage the negative edges to generate negative examples, which necessitates new contrastive objective based on graph information. In this work, we leverage graph level representation in contrastive learning to learn global structure of the product relationship.

3 HETEROGENEOUS GRAPH INFOMAX METHODOLOGY

3.1 Problem Formulation

We denote the product graph by $G = (N, E)$, where the node set of the graph N represents the set of products, with the initial product embeddings $\{\boldsymbol{x}_i\}_{i\in[N]}$ being nodes, and the set of edges E. The graph G is heterogeneous with K edge types corresponding to different product connections, e.g., co-views and co-purchases. We consider an encoder $\mathcal{E}(X, W) : \{\boldsymbol{x}_i\}_{i\in[N]} \rightarrow \{\boldsymbol{z}_i\}_{i\in[N]}$ that is a graph neural network, where $W \in \mathbb{R}^{M\times L}$ is the set of parameters in the encoder, where M is the dimension of the node representation, and L is the number of layers of the encoder. The encoder $\mathcal{E}$ transforms the input node representations to a set of new representations that can be used for downstream tasks. Our goal is to train a set of parameters W for the encoder $\mathcal{E}$ by using multiple learning tasks with different objective functions that learn from different data sources. The parameters W will then be used to generate product embeddings for substitute candidate generation. Given a query product, its substitute candidates can be generated by ranking the products via their output representations and setting a threshold k for its k nearest neighbor (kNN) products.

3.2 HGI Modeling

We propose a multi-task learning model that jointly learns through supervised or unsupervised approaches; the supervised tasks learn from labels collected from different sources of substitutable and non-substitutable signals, and the unsupervised objective learns implicit product relations by extracting information from the graph structure. Our model contains three representative learning tasks, learning from i) customer behavior data, ii) graph structure, and iii) product static information, respectively.

3.2.1 Behavior Learning Task. Customer behavior data is collected from different data streams, including both substitute and non-substitute signals. For example, co-views, which is treated as substitute signals, are the product pairs that are viewed together by customers. Likewise, search pairs, also treated as substitute signals, are the product pairs that are usually clicked together in the same search query. In contrast, co-purchases, the product pairs that are purchased together, should be treated as non-substitute signal since customers rarely buy substitutable products together. For labels, we denote a product pair (N_i, N_j) as a substitute label, where N_i is a query product and N_j is labeled as substitutable to N_i.

This learning task learns from S sources of behavior data, with the loss function being defined as a combination of losses from all data sources. As in [11], we consider two types of loss functions: i) Triplet loss, which captures the pairwise correlation, is used for sparse data sources, i.e., a small average number of substitutable labeled products for each query product, and for each label pair (N_i, N_j), its triplet loss is defined as follows:

$$L_T(\boldsymbol{z}_i, \boldsymbol{z}_j) = \mathbb{E}_{k\sim N} \max\{0, \boldsymbol{z}_i \cdot \boldsymbol{z}_j - \boldsymbol{z}_i \cdot \boldsymbol{z}_k + \Delta\}, \tag{1}$$

where $\boldsymbol{z}_i$, $\boldsymbol{z}_j$, $\boldsymbol{z}_k$ are the output representations of product N_i, N_j, N_k from the encoder $\mathcal{E}$, respectively, and Δ denotes the margin hyper-parameter; ii) Listwise loss is used for denser data sources, i.e., a large average number of substitutable labeled products for each query product. For each query product N_i, there exists a substitute/non-substitute label list $\{N_{i1}, \ldots, N_{iM}\}$ of length M, associated with an indicator list $\{Y_{i1}, \ldots, Y_{iM}\}$ indicating substitute/non-substitute, where the product pairs are sorted in decreasing order in cosine similarity. Then the listwise loss for query product N_i is defined as follows:

$$L_{LW}(\boldsymbol{z}_i, \{\boldsymbol{z}_i\cdot\}, \{Y_i\cdot\}) = \sum_{m=1}^{M} v_m \cdot qAP(\boldsymbol{z}_{im}, Y_{im}),$$

where v_m is a normalized attention weight for the position m in the label list, and $qAP(\cdot)$ is a quantized average precision loss that

is maximized directly using stochastic optimization. The definition of *qAP* loss can be found in Eq. (3) in [11]. The loss function for the behavior learning task is given by $L_B = \sum_{s \in S} \lambda_s \cdot L_s$, where λ_s is the weight for data source s, and L_s is the loss function for the data source s, which could be either a triplet or a listwise loss.

3.2.2 Contrastive Learning Task. In this learning task, we learn from graph structure to update product representations, i.e., learning the product representation from its neighborhood in an unsupervised manner. Inspired by [21], our approach is to define a contrastive objective function, where we minimize the distance between the query node representation and its positive neighbors, and maximize the distance between the query node representation and its negative neighbors. Given the heterogeneous product graph G, we filter the K edge types and obtain a set of positive edge types denoted by K^+, implying positive product relation regarding substitutability (e.g., co-views). For a query node N_i, to obtain its positive neighborhood representation, we randomly sample P levels of neighbors along positive edge types, where each level is denoted by a set $N_{i,p}^+$ with $p \in P$. Then we generate the positive neighborhood representation $\tilde{z}_i^+$ via a readout function $R : \{z_{i,p}\}_{p \in P} \rightarrow \tilde{z}_i$. The distance of a positive relation can be the distance of the query node representation z_i and its neighborhood representation $\tilde{z}_i^+$. For negative relations, we use one of two methods below;

1. Nodes relocation. This method was proposed in [21], where one relocates all the input node representations before the encoder transformation, i.e., shuffling node representation, to generate a negative graph example G^-. Then, given a query node N_i, we obtain its negative query node representation z_i^- through the same encoder $\mathcal{E}(G^-, W)$. Then, with $\tilde{z}_i^+$ and z_i^-, our loss function is defined as follows:

$$L_C(z_i, \tilde{z}_i^+, z_i^-) = \mathbb{E}\left\{ \log \sigma\big(d(z_i, \tilde{z}_i^+)\big) + \log \left[1 - \sigma\big(d(z_i^-, \tilde{z}_i^+)\big) \right] \right\}, \quad (2)$$

where $d(\boldsymbol{a}, \boldsymbol{b})$ denotes the Euclidean distance of vectors $\boldsymbol{a}$ and $\boldsymbol{b}$.

2. Negative neighbor sampling. For heterogeneous graphs with edge types that represents negative node relations, e.g., co-purchases, we propose a new method to obtain the negative representation distance. Similarly to Nodes relocation, given a query node N_i, we randomly sample P levels of neighbors along negative edge types, where each level is denoted by set $N_{i,p}^-$ with $p \in P$. Then we generate the negative neighborhood representation $\tilde{z}_i^-$ via the readout function R. Next, we generate the negative distance by measuring the distance of the query node representation z_i and its neighborhood representation $\tilde{z}_i^-$. Formally, we define our loss function as follows:

$$L_C(z_i, \tilde{z}_i^+, \tilde{z}_i^-) = \mathbb{E}\left\{ \log \sigma\big(d(z_i, \tilde{z}_i^+)\big) + \log \left[1 - \sigma\big(d(z_i, \tilde{z}_i^-)\big) \right] \right\}. \quad (3)$$

3.2.3 Static Information Learning Task. This learning task learns from labels based on static product features, e.g., product images and product titles. Since behavior learning task optimizes product representations regardless of static product features, it is possible that the predicted substitutable products have different appearances or titles, which in some cases conflicts with substitutable criteria in common sense. To this end, we add static information similarity as a separate task, so that we can avoid the situation where the optimization is dominated by learning behavior data, and we can jointly learn content similarities. In this learning task, we generate labels from one data source, which contains labeled product pairs sharing similar product attributes, e.g., product images and titles. We utilize the triplet loss in Eq. (1) as our loss function in this learning task.

To integrate all the learning task objectives in training process, we aggregate the computed gradients from three learning tasks by a weighted sum to specify trade-off's between the tasks, which allows us to generate product representations that reflect our preference over the trade-off's: $L_B + \lambda_C L_C$. For example, if we want to have more strict optimization in learning global patterns than behavior learning, we can put higher weights on the contrastive loss.

Table 1: Approximate training time for each learning task, with 30 epochs and batch size 64.

Learning Task	Training Time
Behavior Learning Task	5 hr
Contrastive Learning Task (Nodes Relocation)	37 hr
Contrastive Learning Task (Negative Neighbors Sampling)	46 hr
Static Info Learning Task	32 min

Table 2: Comparison of two contrastive methods on two learning tasks with different task weights, where HGI-relo is the setting of behavior task and contrastive task with nodes relocation, and HGI-neighbor is the setting of behavior task and contrastive task with negative neighbors sampling.

	Weight	Browse Node	Behavior	
	Ratio	MAP@100/500/1K	NDCG@5/30/500	MAP@5/30/500
Embedding		.8746 .7953 .7566	.0541 .0852 .1438	.0343 .0452 .0507
HGI-relo	10:1	.9622 .9231 .9011	.0932 .1656 .2726	.0614 .0874 .1001
HGI-relo	1:1	.9752 .947 0 .9299	.0847 .1528 .2614	.0555 .0790 .0915
HGI-neighbor	10:1	.9624 .9247 .9026	.0844 .1468 .2547	.0552 .0774 .0894
HGI-neighbor	1:1	.9778 .9523 .9369	.0801 .1465 .2541	.0521 .0748 .0870

4 EXPERIMENTS

4.1 Experimental design

We collect customer behavior data and static product feature from Amazon proprietry data. The behavior data consists of three data sources: CSS, Search logs, Out-Of-Stock logs, where CSS includes co-views, co-purchases, and view-to-purchases, in the format of product pairs. For product static data, we extract images and titles, which are generated by pre-trained Xception (2048 dimension) [4], and by pre-trained BERT (768 dimension) [5], respectively. Both are reduced to 100 dimensions by PCA and concatenated to be an input representation.

We construct the heterogeneous graph with nodes by product representations and edges by behavior data. For fair comparison with [11], we construct the graph with six edge types, including co-views, co-purchases, view-to-purchases, purchase-from-views, search, and OOS. All edge types are bi-directed except view-to-purchases and purchase-from-views. The graph contains ~1M product nodes with their embeddings. Graph details can be found in Table 1 in [11]. We use the heterogeneous graph neural network in [28] as our encoder, which is implemented with the DGL library [22]. Unless stated otherwise, we set batch size to be 64 and learning rate to be 0.0001 for all methods. For a fair comparison, we use the same graph structure and initialization for all methods.

Table 3: Comparison of HGI and its baselines, where all the baselines are the one learning task version of our model that focuses on single objective optimization.

	Browse Node	Behavior		Static Info	
	MAP@100/500/1000	NDCG@5/30/500	MAP@5/30/500	NDCG@5/30/500	MAP@5/30/500
Embedding	.8746 .7953 .7566	.0541 .0852 .1438	.0343 .0452 .0507	.1274 .1812 .2459	.0912 .1112 .1180
M-HetSAGE	.9574 .9151 .8909	.0875 .1543 .2611	.0573 .0812 .0934	.1164 .1730 .2478	.0831 .1032 .1110
DGI	.9812 .9574 .9415	.0544 .0944 .1643	.0346 .0484 .0561	.0975 .1534 .2312	.0688 .0879 .0963
HGI-contrastive	.9524 .8911 .8547	.0368 .0641 .1176	.0231 .0319 .0371	.0588 .0977 .1650	.0401 .0524 .0589
Static Info	.9242 .8687 .8399	.0603 .0976 .1663	.0386 .0518 .0587	**.1282 .1912 .2723**	**.0919 .1148 .1238**
HGI-relo	.9825 .9598 .9454	**.0873 .1571 .2626**	**.0576 .0817 .0938**	.1237 .1906 .2711	.0883 .1124 .1217
HGI-neighbor	**.9859 .9663 .9528**	.0808 .1452 .2475	.0530 .0751 .0866	.1094 .1764 .2588	.0768 .1002 .1096

For the behavior learning task, we generate the labels by filtering the customer behavior data. Specifically, for data source CSS, we use the overlapped pairs from co-views and view-to-purchases with co-purchases being removed to be the labeled positive pairs; for data sources search and OOS, we label product pairs with higher substitutability scores (i.e., search pairs with click score over 50 and OOS pairs with purchase rate over 0.2) as positive pairs. During training process, for each query node, we randomly sample nodes from the graph as its negative labels to compute loss.

For the static information learning task, we generate labels from a similarity based dataset that is based on static (i.e., product images and titles) information whose quality is maintained by human experts. Similarly to the behavior learning task, we randomly sample nodes from the graph as negative labels for query nodes to compute loss. For both behavior and static information learning tasks, we split the label set with 80% for training and 20% for testing.

For contrastive learning task, we set co-purchases edges as negative edges, and all the other edge types as positive edges. For both positive and negative neighbors sampling, we sample three layers of neighbors, each layer with 10 random nodes from positive and negative edge types, respectively. In order to mitigate the conflict of different objective functions, we sample positive neighbors of query nodes along positive edges filtered by positive training labels. We use a simple mean function as readout function, and Euclidean distance for contrastive loss.

The evaluation process is as follows: after we collect all output product representations, we run kNN of a set of query products. Then, we evaluate their Normalized Discounted Cumulative Gain (NDCG) and Mean Average Precision (MAP) with different k on testing labels from behavior, and static info. To evaluate the effect of contrastive loss, we use the browse node, which is the smallest product category in Amazon as a proxy to capture global similarity; Products sharing the same browse node indicate that the products have the same type – hence, it can be considered as a superset of substitutes. The approximate training times for learning tasks are summarized in Table 1.

4.2 Results and Discussions

4.2.1 Comparison of Two Contrastive Methods. We first compare HGI with two learning tasks: behavior and contrastive, and test each of them under two different task weights. HGI-relo, HGI-neighbor are two-task versions of HGI with nodes relocation and negative neighbor sampling, respectively. The results are listed in Table 2, where the row of embedding is the evaluation result on raw input node embeddings. We observe that both contrastive methods clearly outperform the baseline setting on raw embeddings. Comparison between HGI-relo and HGI-neighbor shows trade-off; while HGI-neighbor performs better on browse node, HGI-relo works better on behavior data, and vice versa. As expected, task weights can control balance of the tasks; when putting more weight on behavior learning task, the performance on behavior data will get better on both HGI settings, and correspondingly, the performance on browse node will shrink a bit on both HGI setting, and vice versa.

4.2.2 Comparison of HGI and Baselines. In this experiment, we compare HGI with some baseline settings. For HGI, we set all the task weights as $1:1:1$. M-HetSAGE is a one-task learning method focusing on behavior learning [11], DGI is a one task learning method focusing on contrastive learning, which is a customized method of DGI framework in [21], and sharing an identical model structure with the one-task version of our model HGI-relo, HGI-contrastive is a one-task version of our model HGI-neighbor, Static Info is a one-task learning method focusing on static info learning. We list our experiment results in Table 3, where we observe that each baseline setting performs well on its own learning task, e.g., M-HetSAGE is better on behavior data, DGI and HGI-contrastive are better on browse node, and Static Info is better on static info. When combining all three tasks in HGI, the comprehensive performance of HGI-relo over all testing sets outperforms all the baseline settings, implying that such multi-task framework helps improve learning performance on each single task. Besides, our setting HGI-neighbor has the best performance on browse node, implying that our proposed contrastive method performs better on extracting graph structure and learning from neighborhood.

5 CONCLUSION

In this work, we proposed a multi-task graph learning model HGI, which learns product representation from different sources, either supervised or unsupervised, where each learning task has its own objective and purpose. In particular, we proposed an objective for unsupervised graph learning based on neighborhood information and learns from both positive signals and negative signals extracted from the graph structure. The experiments on Amazon dataset empirically demonstrated that our method outperforms all baselines regarding comprehensive performance among all testing sets. As for future works, one direction is to have better optimization over multi-tasks, potentially by dynamically adjusting learning rates of different tasks during training for a more balanced performance, or by other more sophisticated methods [13, 14, 16]. Another direction is to reduce time complexity of contrastive methods, which is a commonly seen bottleneck for models with graph operations.

REFERENCES

[1] Randolph E. Bucklin, Gary J. Russell, and V. Srinivasan. 1998. A Relationship between Market Share Elasticities and Brand Switching Probabilities. *Journal of Marketing Research* 35, 1 (1998), 99–113. https://doi.org/10.1177/002224379803500110 arXiv:https://doi.org/10.1177/002224379803500110

[2] Tong Chen, Hongzhi Yin, Guanhua Ye, Zi Huang, Yang Wang, and Meng Wang. 2020. Try This Instead: Personalized and Interpretable Substitute Recommendation. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Jul 2020). https://doi.org/10.1145/3397271.3401042

[3] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. 2019. Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks. *CoRR* abs/1905.07953 (2019). arXiv:1905.07953 http://arxiv.org/abs/1905.07953

[4] François Chollet. 2016. Xception: Deep Learning with Depthwise Separable Convolutions. *CoRR* abs/1610.02357 (2016). arXiv:1610.02357 http://arxiv.org/abs/1610.02357

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[6] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. *CoRR* abs/1607.00653 (2016). arXiv:1607.00653 http://arxiv.org/abs/1607.00653

[7] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *CoRR* abs/1706.02216 (2017). arXiv:1706.02216 http://arxiv.org/abs/1706.02216

[8] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation Learning on Graphs: Methods and Applications. *CoRR* abs/1709.05584 (2017). arXiv:1709.05584 http://arxiv.org/abs/1709.05584

[9] Kaveh Hassani and Amir Hosein Khas Ahmadi. 2020. Contrastive Multi-View Representation Learning on Graphs. *CoRR* abs/2006.05582 (2020). arXiv:2006.05582 https://arxiv.org/abs/2006.05582

[10] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 639–648. https://doi.org/10.1145/3397271.3401063

[11] Tong Jian, Fan Yang, Zhen Zuo, Wenbo Wang, Michinari Momma, Tong Zhao, Chaosheng Dong, Yan Gao, and Yi Sun. 2022. Multi-task GNN for substitute identification. In *The Web Conference 2022*. https://www.amazon.science/publications/multi-task-gnn-for-substitute-identification

[12] Fan Liu, Zhiyong Cheng, Lei Zhu, Zan Gao, and Liqiang Nie. 2021. Interest-Aware Message-Passing GCN for Recommendation. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 1296–1305. https://doi.org/10.1145/3442381.3449986

[13] Debabrata Mahapatra, Chaosheng Dong, Yetian Chen, and Michinari Momma. 2023. Multi-Label Learning to Rank through Multi-Objective Optimization. In *Proceedings of the 29th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[14] Debabrata Mahapatra, Chaosheng Dong, and Michinari Momma. 2023. Querywise Fair Learning to Rank through Multi-Objective Optimization. In *Proceedings of the 29th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[15] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring Networks of Substitutable and Complementary Products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Sydney, NSW, Australia) *(KDD '15)*. Association for Computing Machinery, New York, NY, USA, 785–794. https://doi.org/10.1145/2783258.2783381

[16] Michinari Momma, Chaosheng Dong, and Jia Liu. 2022. A multi-objective / multi-task learning framework induced by Pareto stationarity. In *Proceedings of the 39th International Conference on Machine Learning*.

[17] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, Sofus A. Macskassy, Claudia Perlich, Jure Leskovec, Wei Wang, and Rayid Ghani (Eds.). ACM, 701–710. https://doi.org/10.1145/2623330.2623732

[18] Vineeth Rakesh, Suhang Wang, Kai Shu, and Huan Liu. 2019. Linked Variational AutoEncoders for Inferring Substitutable and Supplementary Items. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining* (Melbourne VIC, Australia) *(WSDM '19)*. Association for Computing Machinery, New York, NY, USA, 438–446. https://doi.org/10.1145/3289600.3290963

[19] Allan D. Shocker, Barry L. Bayus, and Namwoon Kim. 2004. Product Complements and Substitutes in the Real World: The Relevance of "Other Products". *Journal of Marketing* 68, 1 (2004), 28–40. https://doi.org/10.1509/jmkg.68.1.28.24032 arXiv:https://doi.org/10.1509/jmkg.68.1.28.24032

[20] Anton Tsitsulin, John Palowitch, Bryan Perozzi, and Emmanuel Müller. 2020. Graph Clustering with Graph Neural Networks. *CoRR* abs/2006.16904 (2020). arXiv:2006.16904 https://arxiv.org/abs/2006.16904

[21] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep Graph Infomax. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rklz9iAcKQ

[22] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. 2020. Deep Graph Library: A Graph-Centric, Highly-Performant Package for Graph Neural Networks. arXiv:1909.01315 [cs.LG]

[23] Zihan Wang, Ziheng Jiang, Zhaochun Ren, Jiliang Tang, and Dawei Yin. 2018. A Path-Constrained Framework for Discriminating Substitutable and Complementary Products in E-Commerce. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (Marina Del Rey, CA, USA) *(WSDM '18)*. Association for Computing Machinery, New York, NY, USA, 619–627. https://doi.org/10.1145/3159652.3159710

[24] Yifan Xing, Tong He, Tianjun Xiao, Yongxin Wang, Yuanjun Xiong, Wei Xia, David Wipf, Zheng Zhang, and Stefano Soatto. 2021. Learning Hierarchical Graph Neural Networks for Image Clustering. *CoRR* abs/2107.01319 (2021). arXiv:2107.01319 https://arxiv.org/abs/2107.01319

[25] Mingyue Zhang, Xuan Wei, Xunhua Guo, Guoqing Chen, and Qiang Wei. 2019. Identifying Complements and Substitutes of Products: A Neural Network Framework Based on Product Embedding. *ACM Trans. Knowl. Discov. Data* 13, 3, Article 34 (jun 2019), 29 pages. https://doi.org/10.1145/3320277

[26] Shijie Zhang, Hongzhi Yin, Qinyong Wang, Tong Chen, Hongxu Chen, and Quoc Viet Hung Nguyen. 2019. Inferring Substitutable Products with Deep Network Embedding. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 4306–4312. https://doi.org/10.24963/ijcai.2019/598

[27] Jiaqian Zheng, Xiaoyuan Wu, Junyu Niu, and Alvaro Bolivar. 2009. Substitutes or Complements: Another Step Forward in Recommendations. In *Proceedings of the 10th ACM Conference on Electronic Commerce* (Stanford, California, USA) *(EC '09)*. Association for Computing Machinery, New York, NY, USA, 139–146. https://doi.org/10.1145/1566374.1566394

[28] Liyuan Zheng, Zhen Zuo, Wenbo Wang, Chaosheng Dong, Michinari Momma, and Yi Sun. 2021. Heterogeneous graph neural networks with neighbor-SIM attention mechanism for substitute product recommendation. In *DLG-AAAI 2021*. https://www.amazon.science/publications/heterogeneous-graph-neural-networks-with-neighbor-sim-attention-mechanism-for-substitute-product-recommendation

[29] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Deep Graph Contrastive Representation Learning. *CoRR* abs/2006.04131 (2020). arXiv:2006.04131 https://arxiv.org/abs/2006.04131

[30] Zhen ZUO, Lixi Wang, Michinari Momma, Wenbo Wang, Yikai Ni, Jianfeng Lin, and Yi Sun. 2020. A flexible large-scale similar product identification system in e-commerce. In *KDD 2020 Workshop on Industrial Recommendation*. https://www.amazon.science/publications/a-flexible-large-scale-similar-product-identification-system-in-e-commerce