Multi-task GNN for Substitute Identification

Tong Jian
jian@ece.neu.edu
Northeastern Univeristy, USA

Fan Yang
fnam@amazon.com
Amazon, USA

Zhen Zuo
zhenzuo@amazon.com
Amazon, USA

Wenbo Wang
wenbowan@amazon.com
Amazon, USA

Michinari Momma
michi@amazon.com
Amazon, USA

Tong Zhao
zhaoton@amazon.com
Amazon, USA

Chaosheng Dong
chaosd@amazon.com
Amazon, USA

Yan Gao
yanngao@amazon.com
Amazon, USA

Yi Sun
yisun@amazon.com
Amazon, USA

ABSTRACT

Substitute product recommendation is important to improve customer satisfaction on E-commerce domain. E-commerce in nature provides rich sources of substitute relationships, e.g., customers purchase a substitute product when the viewed product is sold out, etc. However, existing recommendation systems usually learn the product substitution correlations without jointly considering variant customer behavior sources. In this paper, we propose a unified *multi-task heterogeneous graph neural network* (M-HetSage), which captures the complementary information across various customer behavior data sources. This allows us to explore synergy across sources with different attributes and quality. Moreover, we introduce a *list-aware average precision* (LaAP) loss, which exploits correlations among lists of substitutes and non-substitutes by directly optimizing an approximation of the target ranking metric. On top of that, LaAP leverages a list-aware attention mechanism to differentiate substitute qualities for better recommendations. Comprehensive experiments on Amazon proprietary datasets demonstrate the superiority of our proposed M-HetSage framework equipped with LaAP loss, showing 33%+ improvements on NDCG and mAP metrics comparing to traditional HetSage optimized by a single Triplet loss without differentiating customer behavior data sources.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Substitute Product Identification, Graph Neural Networks

ACM Reference Format:
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 *Companion Proceedings of the Web Conference 2022 (WWW '22 Companion), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3487553.3524247

WWW '22 Companion, April 25–29, 2022, Virtual Event, Lyon, France

ACM ISBN 978-1-4503-9130-6/22/04.
https://doi.org/10.1145/3487553.3524247

1 INTRODUCTION

In E-commerce, recommendation system is of vital importance for suggesting relevant products from a huge candidate pool to meet a variety of user preferences. Despite the tremendous amount of research on improving recommendation systems [1, 13], an important but often overlooked area is learning substitute product correlations [17, 18] from customer feedback. Substitutable products are typically similar, compatible, interchangeable and equally fitting the context of customer's purchase intention [14]. McAuley et al. [10] developed the first method that is capable of predicting substitute products from the reviews and descriptions with topic modeling. More recently, deep learning models are proposed for substitute product identification [2, 11]. Moreover, the success of GNN over a wide range of applications [6, 16] has led to recent applications of GNN on substitute recommendation due to its capability of leveraging both input attributes (content information) and graph structure (behavior data), aiming to provide high-quality embedding representations. In particular, Zheng et al. [19] proposed a heterogeneous GNN (HetSage) architecture to model various customer behavior connections for substitute product recommendation.

However, the aforementioned works focus on using product content information and/or limited user-to-product behavior interactions (e.g., click/browsing logs from recommendation widgets, search pages, etc.) collected from a single source but do not take fully consideration over substitutes generated from different sources. In real scenarios, some sources are accurate but have limited size (e.g., out-of-stock recommendation click data: only available for out-of-stock products); some sources are large-scale but very noisy (e.g., products customer frequently view together: mixed with complementary products besides the substitutable ones). Furthermore, most existing solutions treat substitute recommendation as a pairwise or a triplet learning task. That is, given a query product, the recommendation model learns the most relevant products as substitutes based on a pairwise or triplet loss [2, 15, 19] that enforces non-substitutes discrimination and substitutes similarity. However, one major drawback of triplet loss is that it focuses on the local correlations among triplets, and ignores the global rank of substitutes.

To address the limitations mentioned above, we propose a multi-task heterogeneous GNN (M-HetSage) to further enhance the substitute correlations modeling from multiple types of sources in this paper. In particular, our major contributions include: (i) we propose M-HetSage, a general multi-task GNN framework, which aims

to capture the complementary information across multiple data sources for substitute recommendation; (ii) we propose the List-aware Average Precision (LaAP) loss, which is the first to leverage a listwise loss to directly optimize ranking metrics (i.e., mAP in this paper) for substitute product recommendation; (iii) we experimentally demonstrate the effectiveness of our proposed M-HetSage equipped with LaAP loss on a large-scale Amazon dataset, which shows 33%+ improvement over single-task HetSage model optimized by a single Triplet loss.

2 PROBLEM FORMULATION AND MODELING

In this section, we first formally define the problem and describe three typical data sources in E-commerce to collect potential substitutes. We then present our M-HetSage framework to combine different data sources for multi-task learning. We also describe the proposed LaAP loss in detail, and how we equip M-HetSage with different ranking losses for better recommendation.

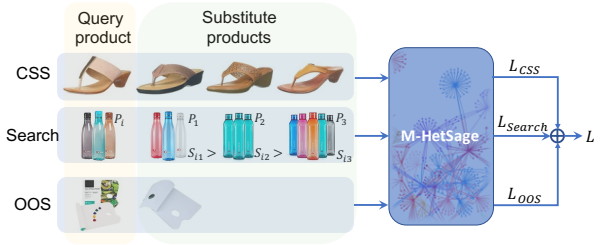

Figure 1: Illustration of our M-HetSage framework for substitute recommendation. We consider three tasks, i.e., CSS, search logs, and OOS, that comprise potential substitute products from different historical customer behavior data sources. M-HetSage unifies three distinct tasks by combining each task-specific loss with a weighted sum.

We denote $\mathcal{D} = \{(P_i, P_j)\}$ as the set of product pairs, labeled by $Y \in \{0,1\}^{|\mathcal{D}|}$ where Y_{ij} is 1 if (P_i, P_j) is a substitute pair and 0 otherwise, respectively. Given a query product P_i, we refer to all of its substitute and non-substitute products as the corresponding *positive* and *negative* samples denoted by P_i^+ and P_i^-.

Data Sources. In this paper, we mainly collect potential substitutable product pairs (P_i, P_j) from three typical data sources in E-commerce domain: (i) CSS: Co-view (customer who views P_i also views P_j), co-purchase (customer who purchases P_i also purchases P_j) and view-to-purchase similarities (customer who views P_i eventually purchases P_j). These relations are collected from recommendation records by ranking products according to the cosine similarity of the sets of users who purchased/viewed them [9]. (ii) Search logs: records of all displayed & clicked products within the same search queries. By aggregating product pairs over search queries and search sessions, each product pair is associated with a substitutability score S_{ij}, measured by the number of co-clicks and click-to-purchases: the higher the S_{ij}, the more likely products in pair (P_i, P_j) are substitutes to each other. (iii) Out-Of-Stock (OOS): click logs of substitute recommendation for OOS products. A substitutability score $S_{ij} = \frac{\#click}{\#view}$ is computed for each pair. Comparing to CSS and search logs, this data source provides the most relevant substitutes for substitute recommendation, but only covers the out-of-stock products, which limits its scale.

M-HetSage Framework. Among GNN models [5, 8], heterogeneous graphs have shown great success on modeling multi-typed product-to-product relations across multiple applications. In this work, we propose M-HetSage, a unified framework for multi-task learning – combining different data sources and loss function types. For a given graph $G_{\mathbf{W}} = (\mathcal{V}, \mathcal{E})$, we model products as nodes $\mathcal{V}$ and K product-to-product relations, $\mathcal{E} = \{\mathcal{E}_1, \ldots, \mathcal{E}_K\}$, as K edge types. As suggested by HetSage [19], let $e_{ij} = (i, j)$ denote the edge that links from the node $i \in \mathcal{V}$ to $j \in \mathcal{V}$. We consider a directed graph, where $e_{ij} = (i, j)$ is different from its reversed edge $e_{ji} = (j, i)$. A node i is a type-k neighbor of node j if and only if $(i, j) \in \mathcal{E}_k$. In addition, each node i is associated with feature $\mathbf{x}_i \in \mathcal{X}$, which represents the product content information (e.g., title, image).

M-HetSage unifies several distinct tasks into a single architecture, by combining several loss functions with a weighted sum. To optimize the output embeddings, we use two ranking losses operating over the same space: Triplet Loss (TL) [19] and the proposed LaAP Loss. We describe each loss function in detail below.

Triplet Loss: Recall that P_i^+ and P_i^- are positive and negative samples for query P_i, and we define $\mathbf{z}_i^+$, $\mathbf{z}_i^-$ and $\mathbf{z}_i$ are their corresponding embeddings, respectively. To optimize the output embeddings for substitute recommendation, we impose a triplet loss penalty, that attains two goals: (i) maximizing the similarity between $\mathbf{z}_i$ and $\mathbf{z}_i^+$ while (ii) minimizing the similarity between $\mathbf{z}_i$ and $\mathbf{z}_i^-$. Formally, we define the triplet loss as follows,

$$L_r = \mathbb{E}_{\mathcal{D}}(\max\{0, \mathbf{z}_i^\top \mathbf{z}_i^- - \mathbf{z}_i^\top \mathbf{z}_i^+ + \Delta\}), \tag{1}$$

where Δ denotes the margin hyper-parameter.

LaAP Loss: Triplet loss only captures pairwise correlation, which lacks the in-group correlation. To resolve this, we include a listwise average precision (AP) loss based on N_i^+ positive and N_i^- negative samples (total: N_i) given query P_i, to capture nuance difference of substitution through ordering:

$$AP(\mathbf{Z}_{i\cdot}, \mathbf{Y}_{i\cdot}) = \frac{1}{N_i^+} \sum_{k=1}^{N_i} C_k(\mathbf{Z}_{i\cdot}, \mathbf{Y}_{i\cdot}) r_k(\mathbf{Z}_{i\cdot}, \mathbf{Y}_{i\cdot}), \tag{2}$$

where C_k is the precision at rank k; r_k is the relevance function defined as $r_k(\mathbf{Z}_{i\cdot}, \mathbf{Y}_{i\cdot}) = \sum_{j=1}^{N_i} Y_{ij} \mathbb{1}[R(\mathbf{Z}_{i\cdot}, k) = j]$; $\mathbf{Z}_{i\cdot} = [Z_{i1}, \ldots, Z_{iN_i}] \in \mathbb{R}^{N_i}$ denotes the cosine similarity between two product embeddings (i.e., $Z_{ij} = \mathbf{z}_i^\top \mathbf{z}_j$) for each list item to the query product P_i; $R(\mathbf{Z}_{i\cdot}, k)$ is the index of the k-the highest value of $\mathbf{Z}_{i\cdot}$; $\mathbf{Y}_{i\cdot}$ denotes the corresponding pair label: Y_{ij} is 1 if (P_i, P_j) is a substitute pair and 0 otherwise, respectively. AP loss given by Eq. (2) cannot be maximized directly using stochastic optimization because the indicator function $\mathbb{1}[\cdot]$ is discontinuous. Thus, we apply a quantized method [12] to approximate AP loss. We define this quantized AP (qAP) loss as follows,

$$qAP(\mathbf{Z}_{i\cdot}, \mathbf{Y}_{i\cdot}) = \frac{1}{N_i^+} \sum_{m=1}^{M} \hat{C}_m(\mathbf{Z}_{i\cdot}, \mathbf{Y}_{i\cdot}) \hat{r}_m(\mathbf{Z}_{i\cdot}, \mathbf{Y}_{i\cdot}), \tag{3}$$

where $\hat{C}_m(\mathbf{Z}_{i\cdot}, \mathbf{Y}_{i\cdot}) = \frac{\sum_{m'=1}^{m} \sum_{i=1}^{N_i} \delta(\mathbf{Z}_{i\cdot}, m')^\top \mathbf{Y}_{i\cdot}}{\sum_{m'=1}^{m} \sum_{i=1}^{N_i} \delta(\mathbf{Z}_{i\cdot}, m')^\top \mathbb{1}}$; $\hat{r}_m(\mathbf{Z}_{i\cdot}, \mathbf{Y}_{i\cdot}) = \sum_{i=1}^{N_i} \delta(\mathbf{Z}_{i\cdot}, m)^\top \mathbf{Y}_{i\cdot}$; $\delta(\mathbf{Z}_{i\cdot}, m) = \max(1 - \frac{(M-1)|\mathbf{Z}_{i\cdot} - b_m|}{2}, 0)$; b_m is the bin

Table 1: Graph and Datasets Statistics. $|\mathbf{P}_i^+|$ is the average number of substitute products for query product P_i.

Graph	# Nodes		# Edges	
M-HetSage	~1M		~85M	
Datasets	# Pairs		Avg. $\|\mathbf{P}_i^+\|$	
	Train	Test	Train	Test
CSS	~2M	~500K	3.74	1.7
search log	~6.3M	~1.6M	23.25	8.39
OOS	~15K	~4K	1.07	1.02
Union	~8.1M	~2.1M	15.17	4.93

Table 2: Single task by HetSage: Compare listwise loss (qAP and LaAP) with triplet loss (TL)

Methods	N^+ N^-	CSS		search log	
		NDCG @5/@30	mAP @5/@30	NDCG @5/@30	mAP @5/@30
TL-2048	1	0.102/0.201	0.077/0.110	0.091/0.152	0.053/0.075
qAP-512	6	0.103/0.204	0.078/0.111	0.095/0.157	0.056/0.079
LaAP-512	6	0.108/0.217	0.081/0.117	0.098/0.164	0.057/0.082
qAP-128	24	0.109/0.215	0.082/0.117	0.095/0.158	0.056/0.079
LaAP-128	24	**0.111/0.223**	**0.084/0.121**	**0.100/0.169**	**0.059/0.084**

centers given positive integer M to partition normalized similarity space $[-1, 1]$ into $M-1$ equal-sized intervals.

We further extend qAP loss by adding an attention mechanism over lists. Recall that some data sources provide a score S for each substitute pair, reflecting the pairwise confidence towards substitution. It is therefore desirable to give greater attention to certain pairs with higher confidence. To exploit nuance knowledge obtained from pairs with different quality, we propose a list-aware attention mechanism on top of the qAP loss. For each given sampled list, we integrate pairwise scores to form a listwise score as its attention weight. Formally, we define our LaAP loss as follows,

$$LaAP(\mathbf{Z}_{i\cdot}, \mathbf{Y}_{i\cdot}) = \frac{1}{B}\sum_{i=1}^{B} w_i \cdot qAP(\mathbf{Z}_{i\cdot}, \mathbf{Y}_{i\cdot}), \tag{4}$$

where w_i is the listwise attention weight: $w_i = \frac{B\cdot\sum_{j=1}^{N_i} S_{ij}}{\sum_{i=1}^{B}\sum_{j=1}^{N_i} S_{ij}}$. B is the factor to ensure $\frac{1}{B}\sum_{i=1}^{B} w_i = 1$. For query product with not enough (i.e., $n < N^+$) positive samples, we sample additional (i.e., $N^+ - n$) negative samples to take place.

To best leverage the local discrimination power brought by triplet-loss and global ranking capability of list-wise losses, we further weight and aggregate the above mentioned losses as a unified loss: $\mathcal{L} = \sum_{t=1}^{T} \lambda_t \cdot L_t$, where λ_t is the balancing parameter over tasks, and L_t is either a triplet loss or LaAP loss.

3 EXPERIMENTS

To evaluate the performance of our proposed M-HetSage empirically, our experiment datasets contain substitute pairs from three different data sources: CSS, search log, and OOS, summarized in Table 1. Specifically, CSS comprises three types of product relationships extracted from historical data: *co-view*, *co-purchase*, and *view-to-purchase*. Therefore, we construct four types of directed edges in the graph, including *co-view*, *co-purchase*, *view-to-purchase*, and *purchase-from-view*. The former two are bi-directed. Similar to [19], we use the overlapped pairs from *co-view* and *view-to-purchase* with *co-purchase* removed to be the labeled positive pairs, and set their substitutability scores as 1 for computing LaAP loss. For search log and OOS, we use all available product pairs to construct graph edges with type of *search* and *oos*, respectively, and label product pairs with higher substitutability scores (i.e., $S \in \mathbb{N} \geq 50$ and $S \in [0, 1] \geq 0.2$ are set for search log and OOS respectively) as positive pairs. For each dataset, we split 80% for training and 20% for testing and construct a *Union* test set by merging all three test sets. During training, for each given query product P_i, we generate samples by uniformly sampling N^+ positive samples from its substitutes set, and N^- negative samples from all products.

For all three datasets, we extract both image and title text feature as the input feature of each product. The visual input is extracted from pre-trained Xception with output size of 2048 [3] and the title text input is extracted by pre-trained BERT with output size of 768 [4]. They are reduced to 100 dimension by PCA respectively and concatenated to become the input product feature.

Following [19], we use a 2-layer GNN, whose hidden layer and output embedding size d' are set to be 1024 and 128 respectively. To reduce the computational cost, we uniformly sample 10 neighbors of each edge type and take its mean to perform neighbor-level aggregation. We use Adam [7] as the optimizer and initialize the learning rate to 0.0001. Unless stated, we set batch size to be $B = 512$ and train model by 90K iterations. For the proposed LaAP, we follow the settings in [12] and explore number of positive N^+ and negative N^- samples as hyperparameters in ablation study. Our models are implemented based on DGL library [1]. For a fair comparison, we use the same graph structure and random weight initialization for all methods. To evaluate the trained embedding representations, we use Normalized Discounted Cumulative Gain (NDCG)@5, 30, and mean Average Precision (mAP)@5, 30.

3.1 Effectiveness of LaAP

We first study the benefits of leveraging the listwise loss with respect to the triplet loss by HetSage model trained on a single task, summarized in Table 2. Specifically, considering that OOS data source only contains about one substitute for each query product in average (see Table 1 for avg. $|\mathbf{P}_i^+|$ of OOS source), we take CSS and search log sources as representatives, since they contain query products with more substitutes, which can more thoroughly test the performance differences of listwise and triplet losses. For each data source in the single task, we train on its training set and test on its corresponding test set. We report NDCG@k and mAP@k for model using triplet (TL) and listwise loss (qAP and LaAP) with batch size B denoted as "$-B$" (i.e., TL-2048 represents model with triplet loss and batch size: 2048). For a fair comparison, we enlarge list length $N = N^+ + N^-$ while reducing batch size to keep each method has similar number of total pairs of substitutes and non-substitutes in each batch (N^+ and N^- are also reported in Table 2). Overall, our proposed LaAP loss (LaAP-128) achieves the best performance over NDCG and mAP metrics for both CSS and search log datasets, by around +10% compared to triplet loss (TL-2048).

[1] https://www.dgl.ai/

Table 3: Evaluation of multi-task learning by M-HetSage.

Row	Methods	$N^{+,-}_{css}$ $N^{+,-}_{search}$ $N^{+,-}_{oos}$	λ_{css} λ_{search} λ_{oos}	Union	
				NDCG @5/@30	mAP @5/@30
[i]	Eval-Feat.	-/-/-	-/-/-	0.054/0.085	0.034/0.045
[ii]	TL-2048	-/-/-	-/-/-	0.088/0.150	0.058/0.081
[iii]	TL-512	1/1/1	0.80/0.20/0.001	0.099/0.173	0.065/0.093
[iv]	LaAP-128	24/-/-	1/-/-	0.104/0.186	0.068/0.100
[v]	LaAP-256	6/6/1	0.75/0.25/0.002	0.107/0.190	0.071/0.101
[vi]	LaAP-128	12/12/1	0.5/0.5/0.002	0.110/0.198	0.073/0.105
[vii]	LaAP-64	24/24/1	0.75/0.25/0.002	**0.112**/**0.203**	**0.075**/**0.108**

LaAP vs. qAP. We observe a clear benefit of adding list-aware attention mechanism: by reinforcing the weight on lists with higher substitutability scores, LaAP outperforms qAP loss by 3%+ and 5%+ on NDCG and mAP metrics for both CSS and search log datasets respectively (comparing LaAP-128 (512) with qAP-128 (512)). This validates our assumption that the higher the substitutability score, the more likely a product pair is substitutable. Thus, the attention list-aware mechanism in our LaAP loss leads to better performance.
Impact of List Length. We also explore the impact of list length in Table 2. Specifically, we enlarge list length $N = N^+ + N^-$ while reducing batch size (see LaAP-128/512). As it can be observed, longer list leads to considerably higher performance, by around 3%+ improvement on both CSS and search log datasets (comparing row LaAP-128 with LaAP-512). This supports our motivation that learning from lists with a listwise loss exploits valuable information across substitute products to the greatest possible extent.

3.2 Effectiveness of Multi-task Learning

As shown in Table 3, we train our proposed M-HetSage on all tasks, and evaluate it on the union test set by NDCG@k and mAP@k (see row [v]-[vii]). The loss weights over tasks (i.e., $\lambda_{\text{css}}, \lambda_{\text{search}}, \lambda_{\text{oos}}$) are selected to achieve best NDCG and mAP. Since OOS provides only one substitute for most query products, we apply triplet loss for OOS instead of listwise loss. We compare with the following settings for reference: Eval-Feat. (row [i]) evaluates the raw embedding before training; TL-2048 (row [ii]) trains a HetSage model over the union of all training sets with a single triplet loss; TL-512 (row [iii]) trains the proposed M-HetSage model but with triplet loss for each task; LaAP-128 (row [iv]) trains a HetSage model on single CSS task with proposed LaAP loss. We adjust batch size with list length to fix the number of pairs in each batch for fair comparison.

We have the following observations. First, with triplet loss, M-HetSage (row [iii]) outperforms HetSage (row [ii]) by 15%+; with proposed LaAP loss, M-HetSage (row [vii]) outperforms HetSage (row [iv]) by 8%+. These indicate that model can benefit more from multi-task setting, which takes advantage of both knowledge sharing and task independence. Second, M-HetSage (LaAP-64) (row [vii]) outperforms M-HetSage (TL-512) (row [iii]) by 15%+, indicating that our proposed LaAP loss can activate the best performance when incorporating with multi-task setting. Third, the setting of balancing parameters over tasks matters. The best performance is achieved by setting $\lambda = [0.75, 0.25, 0.002]$ for LaAP (row [vii]), which puts more attention on CSS than search task while least attention on limited-scale OOS task, achieving more than 33% improvement compared with single-task trained HetSage with triplet loss (row [ii]). The weight for OOS loss λ_{oos} is learned to be small (0.002) due to its limited training data size (=15K) and small list length (=1.07) in average compared with much bigger training data size (=2M, 6.3M) and list length (=3.74, 23.25) of CSS and search log data, which is indicated by Table 1.

4 CONCLUSIONS AND FUTURE WORK

In this paper, we propose M-HetSage, a multi-task learning framework for substitute product recommendation. On top of that, we further propose a LaAP loss, which directly optimizes our target ranking metric mAP equipped with a list-aware attention mechanism. Based on experiments on CSS, search log, and OOS data from an Amazon proprietary dataset, we demonstrate the proposed LaAP loss alone can improve more than 15% on NDCG and mAP comparing to the triplet losses, and our proposed M-HetSage framework equipped with LaAP loss brings 33%+ improvement on NDCG and mAP comparing to a triplet loss based single-task HetSage model without differentiating customer behavior data sources. Our work also encourages the exploration of substitute recommendation beyond the pairwise level, by leveraging a metric that can learn from a list of substitutes. Future directions include exploring a smarter way to adaptively combine loss functions across sources, and additional listwise losses benefiting the substitute recommendation.

REFERENCES

[1] Jesús Bobadilla et al. 2013. Recommender systems survey. *Knowledge-based systems* 46 (2013), 109–132.
[2] Tong Chen et al. 2020. Try This Instead: Personalized and Interpretable Substitute Recommendation. In *SIGIR*. 891—-900.
[3] François Chollet. 2017. Xception: Deep learning with depthwise separable convolutions. In *CVPR*. 1800–1807.
[4] Jacob Devlin et al. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
[5] David K Duvenaud et al. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *NeurIPS*, Vol. 28.
[6] William L Hamilton et al. 2017. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584* (2017).
[7] Diederik P. Kingma et al. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.
[8] Thomas N Kipf et al. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).
[9] Greg Linden et al. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 7, 1 (2003), 76–80.
[10] Julian McAuley et al. 2015. Inferring networks of substitutable and complementary products. In *SIGKDD*.
[11] Vineeth Rakesh et al. 2019. Linked variational autoencoders for inferring substitutable and supplementary items. In *WSDM*.
[12] Jerome Revaud et al. 2019. Learning with Average Precision: Training Image Retrieval with a Listwise Loss. In *ICCV*.
[13] J Ben Schafer et al. 2007. Collaborative filtering recommender systems. In *The adaptive web*. Springer, 291–324.
[14] Allan D Shocker et al. 2004. Product complements and substitutes in the real world: The relevance of "other products". *Journal of Marketing* 68, 1 (2004).
[15] Zihan Wang et al. 2018. A path-constrained framework for discriminating substitutable and complementary products in e-commerce. In *WSDM*. 619–627.
[16] Zonghan Wu et al. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* (2020).
[17] Mingyue Zhang et al. 2016. Complements and Substitutes in Product Recommendations: The Differential Effects on Consumers' Willingness-to-pay. In *IntRS@ RecSys*. 36–43.
[18] Jiaqian Zheng et al. 2009. Substitutes or complements: another step forward in recommendations. In *Proceedings of the 10th ACM Electronic commerce*.
[19] Liyuan Zheng et al. 2021. Heterogeneous Graph Neural Networks with Neighbor-SIM Attention Mechanism for Substitute Product Recommendation.. In *AAAI Deep Learning on Graphs Workshop*.