E-commerce Search via Content Collaborative Graph Neural Network

Guipeng Xv*
Chen Lin†
chenlin@xmu.edu.cn
School of Informatics, Xiamen University
Xiamen, China

Wanxian Guan
Jinping Gou
Xubin Li‡
lxb204722@alibaba-inc.com
Alibaba Group
Hangzhou, China

Hongbo Deng
Jian Xu
Bo Zheng
Alibaba Group
Hangzhou, China

ABSTRACT

Recently, many E-commerce search models are based on Graph Neural Networks (GNNs). Despite their promising performances, they are (1) lacking proper semantic representation of product contents; (2) less efficient for industry-scale graphs; and (3) less accurate on long-tail queries and cold-start products. To address these problems simultaneously, this paper proposes **CC-GNN**, a novel **C**ontent **C**ollaborative **G**raph **N**eural **N**etwork. Firstly, **CC-GNN** enables content phrases to participate explicitly in graph propagation to capture the proper meaning of phrases and semantic drifts. Secondly, **CC-GNN** presents several efforts towards a more scalable graph learning framework, including efficient graph construction, MetaPath-guided Message Passing, and Difficulty-aware Representation Perturbation for graph contrastive learning. Furthermore, **CC-GNN** adopts Counterfactual Data Supplement at both supervised and contrastive learning to resolve the long-tail/cold-start problems. Extensive experiments on a real E-commerce dataset of 100-million-scale nodes show that **CC-GNN** produces significant improvements over existing methods (i.e., more than 10% improvements in terms of several key evaluation metrics for overall, long-tail queries and cold-start products) while reducing computational complexity. The proposed components of **CC-GNN** can be applied to other models for search and recommendation tasks. Experiments on a public dataset show that applying the proposed components can improve the performance of different recommendation models.

CCS CONCEPTS

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

KEYWORDS

E-commerce search, graph neural networks, graph contrastive learning, long-tail problem, cold-start problem

ACM Reference Format:
Guipeng Xv, Chen Lin, Wanxian Guan, Jinping Gou, Xubin Li, Hongbo Deng, Jian Xu, and Bo Zheng. 2023. E-commerce Search via Content Collaborative Graph Neural Network. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3580305.3599320

1 INTRODUCTION

E-commerce search refers to the search module in an E-commerce platform to help customers find the products they desire. It has become an indispensable component and has driven the rapid growth of numerous E-commerce platforms, such as Amazon [23, 30], Alibaba [45], eBay [42], and so on. Therefore, E-commerce search has been extensively studied in academia and industry [1, 3, 4, 6, 18, 20, 22, 23, 42, 45].

Retrieval models generally have an encoding phase, which represents queries and products as low-dimensional embedding vectors. Recently, an increasing number of retrieval models [3, 6, 8, 15, 19–22, 46, 48] rely on Graph Neural Networks (GNNs) [28] in the encoding phase. There are two advantages of using a GNN-based retrieval model: (1) *Convenience of modeling relationships for different E-commerce search problems.* E-commerce search can take textual keywords [1, 4, 18, 22], images [5, 31, 45], or even products [20] as queries. Regardless of the query types, graph is a natural way to model the relationships among queries and products. (2) *Promising performance in E-commerce search with sparse feedback.* E-commerce has insufficient feedback signals to guide the training of retrieval models. GNNs have shown great performance in this situation because they can embed high-order proximity in node embeddings by repeatedly aggregating information from neighboring nodes [14, 36].

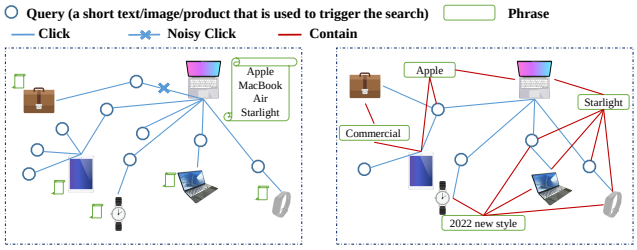
However, E-commerce search has long suffered from the following three challenges, which are not fully solved by recent GNN-based models, especially for data at an industrial scale.

C1: semantic representation of product content. Learning semantic representation of product content (e.g., the title) is crucial, because it is a valuable source to measure query-product relevance. The problem is that the semantics of phrases in the content are usually collaboratively defined by users and merchants on the E-commerce platforms. For example, the exact meaning of the phrase "2022 new style" depends on what products it is used to describe. Furthermore, the meanings of phrases can be changed (i.e., semantic

*Work done during internship at Alibaba Group.
†Corresponding author. Supported by the National Key R&D Program of China (No. 2022ZD0160501), the Natural Science Foundation of China (No. 61972328), Alibaba Group through Alibaba Innovative Research program.
‡Corresponding author.

KDD '23, August 6–10, 2023, Long Beach, CA, USA

ACM ISBN 979-8-4007-0103-0/23/08…$15.00
https://doi.org/10.1145/3580305.3599320



(a) Conventional graph of queries and products

(b) Content Collaborative Graph of queries, products, and phrases

Figure 1: Different graph constructions for E-commerce search. (a) Graph built solely from user feedback, where product contents are treated as node attributes [24, 48]. (b) Content Collaborative Graph presented in this paper, where carefully extracted phrases are nodes explicitly involved in graph propagation.

drift), and it is significantly influenced by correlated products. For example, if MacBook Air uses "starlight" (original meaning: light that comes from the stars) to describe a silver color, other electronic gadgets in silver color will tend to use "starlight" to describe themselves. Most existing studies, as shown in Fig. 1(a), use the content as node attributes and do not update their semantic representations according to the E-commerce data, leading to sub-optimal search performance.

C2: learning efficiency for industry-scale graph. In particular, we focus on two efficiency bottlenecks. (1) Due to the massive volume of nodes and links, only a limited neighborhood of each node can be sampled for message passing [11]. Previous studies [6, 8, 15, 19, 21, 22, 46, 48] do not incorporate link characteristics in this process. Thus their learning efficiencies are restricted by noisy neighbors (e.g., irrelevant neighbors introduced by noisy clicks as shown in Figure 1(a)). (2) Recent studies [38, 40] have strengthened retrieval models by combining supervised learning with self-supervised contrastive learning. Unfortunately, they are impractical for industry-scale graphs because of the expensive graph augmentation to construct positive samples.

C3: long-tail queries and cold-start products. As shown in Figure 1(a), only a few nodes have high degrees, while a large number of nodes have low degrees. Low-degree nodes include long-tail queries, which users infrequently search [16, 25], and cold-start products, which land on the market for a short period of time [10, 39]. These low-degree nodes do not have enough links for conventional GNNs [28] to obtain meaningful neighborhood aggregation, so their representations are inaccurate. This problem hampers the overall search performance and raises fairness issues for customers and online merchants.

These three challenges interact and compound with each other, requiring simultaneous solutions. Thus, we propose a novel **C**ontent **C**olla- borative **G**raph **N**eural **N**etwork (**CC-GNN**) for E-commerce search. **CC-GNN** constructs a Content Collaborative Graph (Figure 1(b)) to jointly model product content and user feedback. The phrase nodes are explicitly involved in the graph propagation to learn their proper semantic representations. **CC-GNN** proposes MetaPath-guided Message Passing, which helps the GNN capture each node's robust topological features. Regarding the learning paradigm, **CC-GNN** combines supervised and contrastive learning. In supervised learning, **CC-GNN** incorporates counterfactual supplement samples for long-tail queries and cold-start products, which are possible training samples "if the queries/products are not long-tail/cold-start". In contrastive learning, to speed up computation, **CC-GNN** presents Difficulty-aware Representation Perturbation, which has the lowest computation complexity w.r.t. SOTA graph contrastive learning methods. **CC-GNN** further enhances the training for long-tail queries and cold-start products by counterfactual data supplement at contrastive learning, which appropriately distinguishes training samples "if the clicks are not affected by popularity bias".

In summary, our contributions are four-fold:

(1) A new graph representation for E-commerce search problem and its efficient construction, to improve search performance by better capturing semantics of content phrases, while reducing storage cost.
(2) Improvements towards a more scalable graph learning framework, in message passing and contrastive learning, which improve performance with lower computational complexity.
(3) Counterfactual data supplement at both supervised and contrastive learning to resolve the long-tail/cold-start problems.
(4) Extensive experiments that verify the superiority of **CC-GNN** on large-scale E-commerce search (more than 100 million nodes). We also empirically show that several proposed components of **CC-GNN** can be flexibly stacked on various models (including GNNs and recommendation models) and boost their performance on different datasets.

2 RELATED WORK

Types of E-commerce search. Depending on the query types, E-commerce search falls into three categories. (1) Conventionally, the search engine takes a short text entered by users as a query [1, 4, 22]. (2) Visual search uses image queries [5, 31, 45] to retrieve products with the same appearance. (3) Many large E-commerce platforms such as Taobao and JD.com have encouraged using products as queries [20]. Product queries are more complex than textual and image queries. A product query provides rich data (e.g., the product's title, description, images, user feedback, etc.) to depict the underlying user preferences in many aspects, including the desired function, price, quality, appearance, and so on. However, E-commerce search with product queries has been rarely explored in the research community. In industry, search engines usually treat product search as a keyword or visual search task.

Graph Representation learning in E-commerce search. Due to the power of graphs in capturing relationships, numerous studies have modeled the E-commerce ecosystem as a large graph and encoded node embeddings by Graph Neural Networks that boost the neural e-commerce search to a new level [8, 20–22, 41]. Most of them build a *heterogeneous graph*, e.g., a bipartite graph of queries and items [15, 19, 46, 48], or a tripartite graph of users, queries, and items [3, 6, 18], from historical *user feedback*. However, the content of items is treated as node attributes and does not involve in the propagation of node embeddings in these works. Furthermore, user feedback is known to be biased and noisy, leading to popularity bias/cold start problems [2]. Some studies build a *homogeneous graph* of items based on content similarity[37, 44]. However, the

E-commerce ecosystem easily reaches the size of millions of items. It is time-consuming to compute pair-wise similarity.

Remarks. In this work, we focus on the under-explored E-commerce search problem with product queries. The proposed Content Collaborative Graph is a heterogeneous graph of queries, products, and phrases nodes. Such a graph has three advantages. (1) It leverages information from the supplier end and supplements missing clicks. (2) The embeddings of phrase nodes are updated in graph propagation, which is more accurate in capturing semantic representations. (3) It does not require time-consuming pairwise content similarity computation. Compared with the recent work in cross-domain recommendation [41], where a single word is a node, our method extracts phrases from the content as nodes. Thus our method is more scalable as each node connects to fewer neighbors. The proposed Content Collaborative Graph Neural Network is tailored for Content Collaborative Graph, which combines supervised and contrastive learning.

3 METHODOLOGY

3.1 Preliminaries

We focus on a particular type of E-commerce search where users use products to search products. The goal of E-commerce search with product queries is to return a list of relevant items for a given query. Note that although a query is also an item that is drawn from the universe of products, we model queries and items separately. In addition to queries and items, we define another type of entity, i.e., phrases which appear in the titles of the queries and items. It is natural to model the three types of entities and their relationships as a graph.

As shown in Figure 2, the proposed method **CC-GNN** is founded on Content Collaborative Graph, formally defined as a heterogeneous graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where its nodes $\mathcal{V} = \{\mathcal{Q}, \mathcal{T}, \mathcal{P}\}$ are a set of query nodes $\mathcal{Q}$, item nodes $\mathcal{T}$ and phrase nodes $\mathcal{P}$. The edges also include three types of edges, a *click edge* $q - t, q \in \mathcal{Q}, t \in \mathcal{T}$ between a query q and a clicked item t, a *query content edge* $q - p, q \in \mathcal{Q}, p \in \mathcal{P}$ between a query q and a phrase p in q's title, and an *item content edge* $t - p, t \in \mathcal{T}, p \in \mathcal{P}$ between an item t and a phrase p in t's title.

The overall framework of **CC-GNN** is shown in Figure 2. We construct Content Collaborative Graph (Section 3.2) and use MetaPath-guided Message Passing to get the node embeddings (Section 3.3). **CC-GNN** combines supervised learning (Section 3.4) and contrastive learning, where the contrastive learning is based on Difficulty-aware Representation Perturbation (Section 3.5) and Counterfactual Data Supplement at contrastive learning (Section 3.6).

3.2 Content Collaborative Graph Construction

To construct the Content Collaborative Graph (CCG), we add click edges $\{q - t\}$ to queries and items. We also need to extract phrases from products' titles. This is a non-trivial problem because computational efficiency and semantic informativeness should be considered. On the one hand, the naive choice of using a word as a node [41] is problematic. Since some popular words (e.g., "new") are used extremely frequently, the graph will have hub nodes that connect to too many neighbors, resulting in unaffordable storage and computation cost. On the other hand, enumerating all phrases is unnecessary. It will hinder information propagation by adding phrases that do not contribute a complete and clear piece of semantic information. Motivated by the above reasoning, we extract informative phrases of indefinite length. There are two steps: *candidate phrase generating* and *phrase pruning*.

3.2.1 Candidate Phrase Generating. Intuitively, informative phrases will be commonly used by correlated products, e.g., to describe a certain desirable product feature. Thus we generate candidate phrases from historical interaction logs.

We collect pairs of queries and items $\{< q, t > | \exists (q - t) \in \mathcal{E}\}$, pairs of co-clicked queries $\{< q, q' > | \exists (q - t) \in \mathcal{E}, (q' - t) \in \mathcal{E}\}$, and pairs of co-clicked items $\{< t, t' > | \exists (q - t) \in \mathcal{E}, (q - t') \in \mathcal{E}\}$. We extract N-gram phrases that appear in both titles of more than three pairs. To increase coverage (i.e., to find as many informative phrases as possible), we do not limit the length of phrases. Since phrases with the same words but different word orders usually have equivalent or similar roles in content collaboration, to avoid redundant sequence comparisons, we rewrite each phrase and reorder the words in a phrase in alphabetical order. For example, "fine wine glasses" and "wine glasses fine" will be rewritten as "fine glasses wine".

3.2.2 Phrase Pruning. To further reduce the size of phrases and accelerate graph computation, we filter out phrases that are too vague and irrelevant to any product feature. We use an E-commerce Named Entity Recognition (NER) tool to identify and score the entities in each candidate phrase. If the total score of the phrase falls below a predefined threshold ξ_s, the phrase will be deleted.

3.2.3 Feature Transformation. Queries and items are associated with sparse features and content features. (1) Sparse features (e.g., prices, sales, cities) are segmented and transformed into one-hot vectors. Suppose there are K fields, i.e., $\mathbf{f}_1, \cdots, \mathbf{f}_K$. (2) Content features include a title and a product image. We use a pre-trained language model to transform the title into a vector $\mathbf{f}_t$, and a pre-trained visual model to transform the product image into $\mathbf{f}_v$.

Thus we initialize the node embeddings from the concatenation of all features.

$$\begin{aligned} \mathbf{q}^0 &= \mathbf{W}^Q \times [\mathbf{f}_1 || \mathbf{f}_2 || \cdots || \mathbf{f}_K || \mathbf{f}_t || \mathbf{f}_v], \\ \mathbf{t}^0 &= \mathbf{W}^T \times [\mathbf{f}_1 || \mathbf{f}_2 || \cdots || \mathbf{f}_K || \mathbf{f}_t || \mathbf{f}_v], \end{aligned} \tag{1}$$

where $\mathbf{q}^0, \mathbf{t}^0$ are the initialized query and item embeddings respectively, $\mathbf{W}^Q, \mathbf{W}^T \in \mathbb{R}^{D \times F}$ are linear transformations, D is the embedding size, F is the feature size, $D \ll F$ for dimensionality reduction, $||$ represents the concatenation of features.

For any phrase node p, we again use a pre-trained language model to obtain the textual feature vectors and transform them into the initial phrase embedding $\mathbf{p}^0$.

$$\mathbf{p}^0 = \mathbf{W}^P \times MeanPooling(\mathbf{f}_{w_1}, \cdots, \mathbf{f}_{w_L}), \tag{2}$$

where $w_i, i = 1, \cdots, L$ is the word in the phrase with length L.

3.3 MetaPath-guided Message Passing

Inspired by GraphSage [11], large-scale graph propagation is achieved by sampling a local neighborhood and aggregating information from the local neighborhood. On Content Collaborative Graph, a dominant fraction of edges is click edges. This means that if we randomly sample the neighborhood, we rely heavily on information propagation through user feedback. However, user feedback can be uncertain and noisy, e.g., a click is the consequence of random

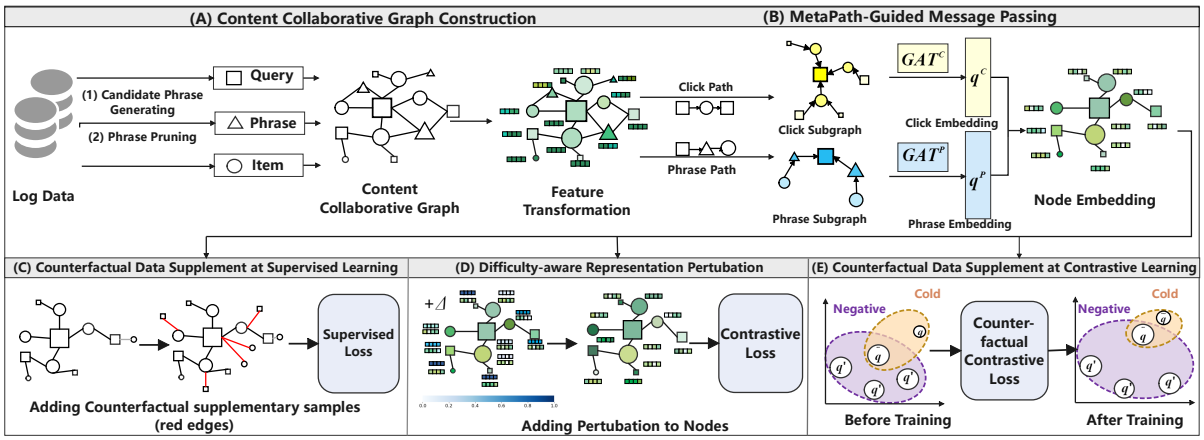


Figure 2: Overall framework of CC-GNN

browsing instead of user preference. If we apply the vanilla sampling, the training will be inefficient. Therefore we adopt MetaPath-guided Message Passing (MPMG).

As shown in Figure 2 (B), we define two metapaths: a *click path* and a *phrase path*. For query nodes, the *click path* is $q-t-q$ and the *phrase path* is $q-p-t$. For item nodes, the *click path* is $t-q-t$, and the *phrase path* is $t-p-q$.

The click path and phrase path guide the sampling process. Let's denote the sub-graph sampled for a query node $q \in \mathcal{Q}$ on click path as $\mathcal{G}^{q,c}$, and the sub-graph sampled on phrase path as $\mathcal{G}^{q,p}$. The two sub-graphs are of equal node size, i.e., for each hop on the path (maximal two hops), we sample K neighbor nodes along the corresponding path. This ensures that content-relevant queries/items will play an important role in message passing.

We adopt two Graph Attention Networks (GAT) [35] as the aggregator on the sampled sub-graphs. The final query embedding is obtained by merging the node embeddings of the sub-graphs.

$$\begin{aligned} \mathbf{q}^c = GAT^c(\mathcal{G}^{q,c}), \quad \mathbf{q}^p = GAT^p(\mathcal{G}^{q,p}), \\ \mathbf{q} = MeanPooling(\mathbf{q}^c, \mathbf{q}^p), \end{aligned} \tag{3}$$

where $GAT(\cdot)$ represents the output of an L-layer GAT with the $LeakyReLU$ function, $\mathbf{q}^c$ represents the query embeddings from $\mathcal{G}^{q,c}$ and $\mathbf{q}^p$ represents the phrase embeddings from $\mathcal{G}^{q,p}$.

Similarly, we can obtain two sub-graphs for the item node and its final item embedding $\mathbf{t}$. Note that we do not sample subgraphs for phrase nodes, but the phrase embeddings are updated during the GAT computations.

3.4 Counterfactual Data Supplement at Supervised Learning

We adopt pair-wise supervised learning to train the model. In a mini-batch $\mathcal{B}$, for each positive pair, i.e., a query and an item that has been clicked under the query (q, t), we sample N negative items t' under the same category of the positive item t and that t' has not been clicked under the query.

Under the supervised learning scheme, long-tail query nodes and cold-start item nodes are under-represented because they lack positive samples in the training batch. To address this problem, we generate counterfactual supplementary samples.

For a long-tail query node $\tilde{q} \in \mathcal{B}$, we first retrieve M similar items $\mathcal{T}^{\tilde{q}} = \{\tilde{t}\}$, based on the cosine similarity on content features, i.e., $cos(\mathbf{f}^{\tilde{q}}, \mathbf{f}^{\tilde{t}}) > \xi_c$, where ξ_c is a predefined threshold. Then, we sample $\lceil M/10 \rceil$ items $\tilde{t}$ from $\mathcal{T}^{\tilde{q}}$ based on its number of clicks, and form $\tilde{\mathcal{B}} = \{(\tilde{q}, \tilde{t})\}$. Thus, we add items that are most likely to be clicked "if the query is not a long-tail query".

Similarly, we can add queries to the mini-batch for a cold-start item $\hat{t}$ by first selecting M similar queries and sampling $\lceil M/10 \rceil$ queries $\hat{q}$ based on the number of clicks. This forms $\hat{\mathcal{B}} = \{(\hat{q}, \hat{t})\}$.

The results of counterfactual data supplement at supervised learning (CDS-SL) are illustrated in Figure 2 (C). For smaller nodes on the graph (i.e., long-tail queries and cold-start items), counterfactual data supplement correlates more nodes with long-tail queries and cold-start items (red edges). Since the counterfactual supplementary samples are not true clicks, we use a confidence level to "downgrade" them in the supervised loss:

$$\mathcal{L}_{sup} = \sum_{(q,t)\in\mathcal{B}\bigcup\tilde{\mathcal{B}}\bigcup\hat{\mathcal{B}}} -\theta_{q,t} log \frac{exp(sim(\mathbf{q},\mathbf{t})/\tau)}{exp(sim(\mathbf{q},\mathbf{t})/\tau) + \sum_{t'} exp(sim(\mathbf{q},\mathbf{t}')/\tau)}, \tag{4}$$

where (q, t) are either true positive pairs or counterfactual supplementary pairs, t' is a negative sample for query q, τ is the temperature, $sim(\mathbf{q}, \mathbf{t})$ is the cosine similarity of node embeddings (Equation 3), confidence level θ is maximal for true positive samples. Otherwise, θ is measured by cosine similarity on the content features.

$$\theta_{q,t} = \begin{cases} 1, & \text{if } (q,t) \in \mathcal{B} \\ cos(\mathbf{f}^q, \mathbf{f}^t), & \text{if } (q,t) \in \tilde{\mathcal{B}} \bigcup \hat{\mathcal{B}}. \end{cases} \tag{5}$$

3.5 Difficulty-aware Representation Perturbation

We combine supervised learning with self-supervised graph contrastive learning, as graph contrastive learning has been successfully applied in E-commerce recommendation and search with sparse labels [8, 38, 43].

The main idea of contrastive learning is to contrast representations of positive (i.e., augmented) and negative samples. Data augmentation, to a large extent, determines the effect of contrastive learning. Augmentation on graph can be carried on the structure (e.g., SGL [38] and GCA [52]), or directly upon embedding space (e.g., SimGCL [43] and COSTA [47]). The former type has larger computational complexity as shown in Table 1. The limitation of current embedding augmenting methods [43, 47] is that they add the same strength of disturbance to all nodes (including low-degree

Table 1: Computational complexity of existing graph contrastive learning methods, E: number of edges, D: embedding size, L: number of convolutional layers and R: dropout rate.

Augmentation Type	Model	Computational Complexity	
		Adjacency Matrix	Graph Convolution
Augmentation on Structure	SGL	$\Theta(2E+4RE)$	$\Theta((2+4R)ELD)$
	GCA	$\Theta(2E+4RE)$	$\Theta((2+4R)ELD)$
Augmentation on Embedding	SimGCL	$\Theta(2E)$	$\Theta(6ELD)$
	COSTA	$\Theta(2E)$	$\Theta(4ELD)$
	DARP	$\Theta(2E)$	$\Theta(2ELD)$

nodes and high-degree nodes). Because high-degree nodes are usually trained well, adding the same amount of disturbance as low-degree nodes will cause unstable training.

Motivated by the above argument, we propose Difficulty-aware Representation Perturbation (DARP). Formally, for each anchor query $\mathbf{q}$, we obtain a positive sample $\bar{\mathbf{q}}$ by adding some disturbance on $\mathbf{q}$. The strength of disturbance is based on the degree of the node, i.e., the normalized number of clicks $NC(q)$ and the normalized number of phrases $NP(q)$.

$$\bar{\mathbf{q}} = \frac{MeanPooling\big(\mathbf{q}+\Delta_q^c \times NC(q), \mathbf{q}+\Delta_q^p \times NP(q)\big)}{||MeanPooling\big(\mathbf{q}+\Delta_q^c \times NC(q), \mathbf{q}+\Delta_q^p \times NP(q)\big)||_2^2}, \tag{6}$$

where $NC(q) = \rho_q^c / \max_q \rho_q^c$ and $NP(q) = \rho_q^p / \max_q \rho_q^p$, ρ_q is the segmentation function which maps the clicks or phrase connections to intervals. $\bar{\mathbf{q}}$ is L_2 normalized.

The disturbance vector Δ_q^x is generated as follows.

$$\bar{\Delta^x} \sim \mathbb{U}(0,1), \quad \Delta_q^x = \bar{\Delta^x} \odot sign(\mathbf{q}), \quad s.t. \|\Delta_q^x\|_2 = \epsilon. \tag{7}$$

where $\bar{\Delta^x}$ a randomly sampled $D-$dimensional vector, $\odot$ is the element-wise multiplication, $sign(\mathbf{q})$ returns a vector indicating the sign of each element in $\mathbf{q}$. $\bar{\Delta^x} \sim \mathbb{U}(0,1)$ obtains a random positive seed convenient for future computation. $\Delta_q^x = \bar{\Delta^x} \odot sign(\mathbf{q})$ transforms the seed into a noise vector which has the same direction as $\mathbf{q}$, ensuring that the perturbation has the same sign as the gradient of the loss function with respect to $\mathbf{q}$. This will result in a smoother training process. $\|\Delta_q^x\|_2 = \epsilon$ controls the magnitude of noise. ϵ is small enough to ensure that the perturbed representation retains most of the original representation's information while maintaining some variance.

The process is shown in Figure 2 (D), the augmentation is generated and added to the node embedding. The smaller nodes get bigger perturbations (i.e., the color of small nodes is changed more significantly after perturbation).

From the minibatch with counterfactual data supplement $\mathcal{B} \bigcup \tilde{\mathcal{B}} \bigcup \hat{\mathcal{B}}$ (defined in Section 3.4), we remove redundant queries and form a set of distinct queries, i.e., $\bar{\mathcal{B}}^q$. The contrastive loss for a query q in this set $\bar{\mathcal{B}}^q$ is

$$\mathcal{L}_{cl}^q = -log \frac{exp(\bar{\mathbf{q}}^T \bar{\mathbf{q}}^{'}/\tau)}{\sum_{g \in \bar{\mathcal{B}}^q} exp(\bar{\mathbf{q}}^T \bar{\mathbf{g}}/\tau)}, \tag{8}$$

where $\bar{\mathbf{q}}$ and $\bar{\mathbf{q}}^{'}$ are two augmentations on the same query q, $\bar{\mathbf{g}}$ is the augmented representation on a negative sample $g \in \bar{\mathcal{B}}^q$. The CL loss encourages consistency between augmented representations of the same query while minimizing the agreement between distinct queries.

Similarly, from the minibatch with counterfactual data supplement $\mathcal{B} \bigcup \tilde{\mathcal{B}} \bigcup \hat{\mathcal{B}}$, we remove redundant items and form a set of distinct items, i.e., $\bar{\mathcal{B}}^t$. For each items t in $\bar{\mathcal{B}}^t$, we apply similar augmentations to item $\mathbf{t}$.

$$\bar{\mathbf{t}} = \frac{MeanPooling\big(\mathbf{t}+\Delta_t^c \times NC(t), \mathbf{t}+\Delta_t^p \times NP(t)\big)}{||MeanPooling\big(\mathbf{t}+\Delta_t^c \times NC(t), \mathbf{t}+\Delta_t^p \times NP(t)\big)||_2^2}, \tag{9}$$

where $NC(t) = \rho_t^c / \max_t \rho_t^c$ and $NP(t) = \rho_t^p / \max_t \rho_t^p$. The contrastive loss for this item is

$$\mathcal{L}_{cl}^t = -log \frac{exp(\bar{\mathbf{t}}^T \bar{\mathbf{t}}^{'}/\tau)}{\sum_{h \in \bar{\mathcal{B}}^t} exp(\bar{\mathbf{t}}^T \bar{\mathbf{h}}/\tau)}, \tag{10}$$

where $\bar{\mathbf{t}}$ and $\bar{\mathbf{t}}^{'}$ are two augmentations on t, $\bar{\mathbf{h}}$ is the augmented negative sample of $h \in \bar{\mathcal{B}}^t$. Then, we define the contrastive loss as:

$$\mathcal{L}_{cl} = \sum_{q \in \mathcal{B}, t \in \mathcal{B}} (\mathcal{L}_{cl}^q + \mathcal{L}_{cl}^t). \tag{11}$$

Computational Complexity Analysis. The computational complexity of graph contrastive learning is strongly affected by two parts, computing the adjacency matrix, and graph convolution [38, 43], as shown in Table 1.

In computing adjacency matrix, Difficulty-aware Representation Perturbation (DARP) has the same computational complexity as other augmentation on embedding methods, and smaller complexity than augmentation on structure methods. Augmentation on embedding methods (i.e., SimGCL, COSTA, and DARP) do not change the original graph structure, so they just need to normalize the adjacency matrix before feeding it to graph convolution, and the complexity is proportional to the number of edges, i.e., $\Theta(2E)$ [43]. For augmentation on structure methods (i.e., SGL and GCA), they have additional computations of dropping out edges with dropout rate R, hence their computational complexity is larger [38].

In graph convolution stage, DARP has the smallest computational complexity. The complexity of graph convolution is related to the number of convolutional layers L, the number of edges E, the embedding size D, and the number of augmented graphs. Each graph has $\Theta(2ELD)$ complexity. (1) for SGL and GCA, they have one original and two augmented graphs, so the complexity is $\Theta((2+4R)ELD)$ [38, 52]. (2) SimGCL adds noise to embeddings in each graph convolutional layer, which means it triples the convolution computation [43]. (3) COSTA uses covariance-preserving feature space augmentation to construct a single-view positive sample, which means it doubles the convolution computation [47]. (4) DARP augments the representations after graph convolution, and does not change the complexity of convolution.

Overall, *DARP has smaller computation complexity than existing graph contrastive methods.*

3.6 Counterfactual Data Supplement at Contrastive Learning

The motivation of Counterfactual Data Supplement at Contrastive Learning (CDS-CL) is to handle long-tail and cold-start problems from the perspective of true and false relevance. We believe that true relevance corresponds to items that are content-relevant and will be clicked under the same query by users. False relevance corresponds to items that are irrelevant, but the model makes a wrong prediction based on a false correlation of feedback. Previous works have mainly focused on learning true relevance and rarely considered false relevance [10, 27, 29, 49]. For example, some works [29, 49] learn shared representations of diverse item attributes. However, as the click feedback is biased toward head items, their attributes tend

to dominate the prediction. This makes it difficult to distinguish the representations of head attributes after training, resulting in inaccurate click predictions when using head attributes. We simultaneously learn the true relevance and eliminate the effect of false relevance. (1) We pull pairs of <warm/head item, cold/tail items> closer in the representation space to learn the true relevance. (2) We push the representations of dissimilar items with the same level of click numbers apart to eliminate the effect of false relevance.

As shown in Figure 2(E), after Counterfactual Data Supplement at Contrastive Learning, the cold item is pulled closer to the warm item and the warm item is moved away from the negative items.

For each cold item $\hat{t} \in \mathcal{B} \bigcup \tilde{\mathcal{B}} \bigcup \hat{\mathcal{B}}$, we first retrieve similar warm items $\dot{t}$ and form a warm item set $\dot{\mathcal{T}}^{\hat{t}}, \forall \dot{t} \in \mathcal{T}^{\hat{t}}, cos(\mathbf{f}^{\hat{t}}, \mathbf{f}^{\dot{t}}) > \xi_c$, where $cos()$ is the cosine similarity on content feature $\mathbf{f}$, and ξ_c is a predefined threshold. Then, we randomly sample one similar warm item $\dot{t}$ from $\dot{\mathcal{T}}^{\hat{t}}$ as a positive sample. $\dot{t}$ should be similar in representation space to $\hat{t}$ "if $\hat{t}$ has been exposed earlier". For each $\dot{t}$ and $\hat{t}$, we sample N negative items t' which have the same number segment of clicks, and that t' are dissimilar. The counterfactual contrastive loss of item is:

$$\mathcal{L}^{t}_{cl_c} = -log\frac{exp(sim(\hat{\mathbf{t}}, \dot{\mathbf{t}})/\tau)}{exp(sim(\hat{\mathbf{t}}, \dot{\mathbf{t}})/\tau) + \sum_{t'} exp(sim(\dot{\mathbf{t}}, \mathbf{t}')/\tau)}, \tag{12}$$

where $\hat{t}, \dot{t}$ are positive samples to each other, t' are the negative samples, τ is the temperature, $sim()$ is the cosine similarity of node embeddings obtained by Equation 3.

Similarly, we can form counterfactual contrastive pairs for queries. For each tail query $\tilde{q}$, we retrieve similar head queries and form a head queries set $\ddot{Q}^{\tilde{q}}$ and randomly sample one head query $\ddot{q}$. We sample N negative queries q' which have the same number segment of clicks, and that q' are dissimilar. The counterfactual contrastive loss of query is:

$$\mathcal{L}^{q}_{cl_c} = -log\frac{exp(sim(\tilde{\mathbf{q}}, \ddot{\mathbf{q}})/\tau)}{exp(sim(\tilde{\mathbf{q}}, \ddot{\mathbf{q}})/\tau) + \sum_{q'} exp(sim(\ddot{\mathbf{q}}, \mathbf{q}')/\tau)}, \tag{13}$$

where $\tilde{q}, \ddot{q}$ are positive samples to each other, q' are the negative samples. The counterfactual contrastive loss is:

$$\mathcal{L}_{cl_c} = \sum_{q \in \mathcal{B}, t \in \mathcal{B}} (\mathcal{L}^{q}_{cl_c} + \mathcal{L}^{t}_{cl_c}). \tag{14}$$

The final loss of **CC-GNN** consists of the supervised loss $\mathcal{L}_{sup}$, contrastive loss $\mathcal{L}_{cl}$, and counterfactual contrastive loss $\mathcal{L}_{cl_c}$.

$$\mathcal{L} = \mathcal{L}_{sup} + \alpha \mathcal{L}_{cl} + \beta \mathcal{L}_{cl_c}, \tag{15}$$

where α, β adjust the losses to the same magnitude.

4 EXPERIMENT

In this section, we study the following research questions:

(1) **RQ1:** Can **CC-GNN** improve the performance of industrial-scale product search (Section 4.2)
(2) **RQ2:** How does each component of **CC-GNN** affect the search performance? Specifically, we perform ablation study on "Content Collaborative Graph Construction" (Section A.3), "MetaPath-guided Message Passing " (Section 4.4), "Difficulty-aware Representation Perturbation " (Section 4.5) and "Counterfactual Data Supplement " (Section 4.6).
(3) **RQ3:** Can **CC-GNN** shed insight into other tasks, e.g., can components of **CC-GNN** be applied to recommendation models and improve their performances (Section 4.7)?

Table 2: Overall ranking results on IPQS

Model	Recall@100	MRR@100	NDCG@100
GraphSAGE	0.4441	0.0819	0.1528
AdaptiveGCN	0.4434	0.0825	0.1531
LasGNN	0.4805	0.1038	0.1759
CC-GNN	**0.5450**	**0.1189**	**0.2046**

4.1 Experimental Setup

Datasets. (1) From Section A.3 to Section 4.6, we use an Industry-scale Product Query Search dataset for evaluation. The dataset, named IPQS, is constructed by collecting 91 days of log data in a real-world E-commerce platform. Each record in IPQS dataset corresponds to a product query and a clicked item and contains the necessary information about the queries and items, including product IDs, categories, prices, sales, images, titles, etc. We randomly sample **35 million** queries, **87 million** items, and **709.7 million** interactions from the log data in the first 90 days as training data. We use the log data of the last day for testing. We remove products (i.e., either queries or items) that do not appear in the training set. (2) For Section 4.7, we evaluate on Amazon Sport Dataset [12, 26], containing **35,598** users, **18,357** items, and **296,337** interactions, which is a widely used recommendation dataset [32, 50, 51]. We pre-process it with a 5-core setting on both items and users following [32, 50, 51].

Evaluation Metrics. We evaluate the ranking quality at top 100 results, i.e., Recall (*Recall*@100), Mean Reciprocal Rank (*MRR*@100), and Normalized Discounted Cumulative Gain (*NDCG*@100). Higher values of *Recall*@100, *MRR*@100, and *NDCG*@100 indicate more accurate search results.

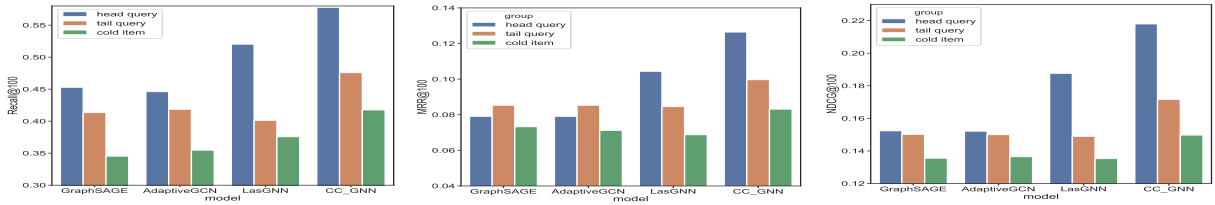
We make the codes of Difficulty-aware Representation Perturbation (DARP) and Counterfactual Data Supplement at Contrastive Learning (CDS-CL) modules of the CC-GNN available [1]. Other implementation details, ablation experiments, and parameter experiments are provided in the appendix.

4.2 Performance on Product Search

Baselines. We compare **CC-GNN** on IPQS dataset with different GNNs in the encoding phase of retrieval models. (1) GraphSAGE [11] samples and aggregates messages from a local neighborhood. GraphSAGE has been widely utilized in retrieval models [33, 34]. (2) AdaptiveGCN [17] samples neighbors and uses the sampled subgraph and learned residual graph to generate node embeddings. (3) LasGNN [9] samples the neighbors layer-wise along the metapaths and constructs a subgraph to aggregate messages. LasGNN has a great advantage in large-scale graph learning and has been used in a real E-commerce production system [8]. Note that due to the massive size of our IPQS dataset, many GNNs in the literature are infeasible. In the matching phase of retrieval models, we apply an Approximate Nearest Neighbor (ANN) search to return results that have the largest cosine similarity to the query in representation space. For these GNN baselines, we use the open source implementations [2]. All the baselines are trained in the supervised loss.

Overall performance Analysis. As shown in Table 2, **CC-GNN** *significantly outperforms all competitors in terms of all evaluation*

[1] https://github.com/XMUDM/CC-GNN
[2] https://github.com/alibaba/euler



(a) *Recall*@100 (b) *MRR*@100 (c) *NDCG*@100

Figure 3: Comparative performance on different groups of queries and items

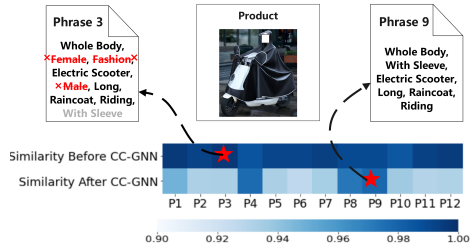


Figure 4: Visualization of the similarity of product and phrase embeddings (P1 to P12) before and after CC-GNN.

metrics. **CC-GNN** improves over the best baseline (i.e., LasGNN) in terms of overall *Recall*@100, *MRR*@100, *NDCG*@100 by 11.8%, 14.5%, and 16.3%, respectively.

Head queries and long-tail queries. To compare the performance of GNNs on low-degree nodes, we divide the queries in the training set into 5 groups according to the number of clicks. Each group of queries has the same query numbers. Queries in the highest click group are head queries, and queries in the lowest click group are tail queries. From Figure 3, we observe a large performance gap between head queries and long-tail queries by all models. Performances of all baselines on long-tail queries are very close. However, **CC-GNN** *significantly improves long-tail query performance*. The *Recall*@100, *MRR*@100, *NDCG*@100 on long-tail queries are increased by 13.7%, 16.7%, 14.2% comparing with the best baseline.

Cold-start items. Given the date of the testing set, we define cold-start items as items that have been on the market for less than seven days. There are more than **45 thousand** cold-start items in the training set, which accounts for 0.2% of all items. We report search performance on cold-start items in Figure 3 and observe that **CC-GNN** *greatly enhances cold-start item performance*. The *Recall*@100, *MRR*@100, *NDCG*@100 on cold-start items are increased by 11.1%, 13.5%, 9.8%, respectively.

Semantic representation of phrase content. Here we provide a showcase to demonstrate that **CC-GNN** can capture the semantics of content phrases. First, we randomly sample one product from the IPQS dataset and collect 12 phrases (i.e., from P1 to P12) that are connecting with the product on Content Collaborative Graph. Then, we extract the embeddings of these phrases before and after **CC-GNN**. Next, we calculate cosine similarity between the product embedding and the phrase embeddings before and after **CC-GNN**.

As shown in Figure 4, before **CC-GNN**, all phrases are highly similar to the product (i.e., > 0.99). Note that since all existing methods do not update the phrase embeddings, this means *all existing methods are not able to distinguish these phrases*. On the contrary, after **CC-GNN**, the differences between phrases become more apparent. It suggests that **CC-GNN** *can adapt the semantic representation of phrase to the E-commerce data*. The most relevant phrase with the highest similarity before **CC-GNN** is Phrase3, and after **CC-GNN** it is Phrase9. Examining these phrases, we can see that Phrase3 contains a very general, noisy term "fashion" that does not describe the property of the product, and is missing the term "with sleeves" which is an important feature that consumers will consider when purchasing the raincoat. Thus, Phrase3 is indeed noisy and Phrase9 is better than Phrase3. **CC-GNN** *can correctly identify the most relevant phrase to describe a product* by learning the phrase semantics in graph propagation.

4.3 Ablation on Content Collaborative Graph

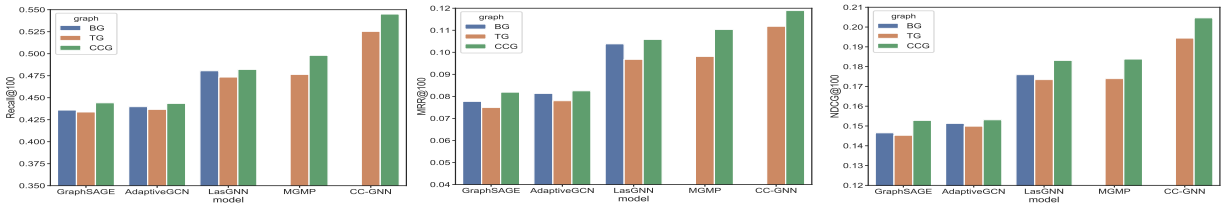
We first investigate whether different graph constructions affect our model's performance and whether Content Collaborative Graph can benefit other graph learning models. Toward this goal, we implement three different graph constructions, including: (1) conventional bipartite query-item graph (BG), which has been used in most search and recommendation problems [15, 19]. The BG graph has **122** million nodes and **710** million edges. (2) Tripartite query-item-phrase graph (TG), which has query nodes, item nodes, and phrase nodes. N-gram phrases are extracted from product titles and added to the graph without pruning. There are **138.4** million nodes and **837** million edges in TG. (3) Content Collaborative Graph (CCG). N-gram phrases are extracted from product titles and added to the graph with pruning. The threshold ξ_s in Section 3.2.2 controls the phrases to be pruned. Here we set the hyperparameter $\xi_s = 6$. CCG contains **128** million nodes and **734** million edges, where **6** million nodes are phrase nodes and **24** million edges are connecting to phrase nodes.

We test three supervised graph learning methods (i.e., GraphSAGE, AdaptiveGCN, and LasGNN) and two model variants (i.e., MGMP and **CC-GNN**) on the different graphs (i.e., BG, TG, and CCG with the hyperparameter $\xi_s = 6$).

Results and Analysis. We have three observations from Figure 5. (1) *For all GNN backbones, the best performance is achieved on* Content Collaborative Graph. The result shows that simultaneously propagating content and collaborative information in a graph neural network improves ranking accuracy. (2) The results on TG are worse than the results on BG for all GNN backbones. This highlights the importance of an appropriate abstract of the relations in the E-commerce ecosystem. It is necessary to prune phrases. Without pruning, many vague phrase nodes or irrelevant phrase nodes will be added, introducing too much noise to the information propagation process. (3) Reasonably, **CC-GNN** obtains the best performance on Content Collaborative Graph, where *Recall*@100, *MRR*@100, *NDCG*@100 are increased by 3.8%, 6.4%, 5.2% comparing to TG.

4.4 Ablation on Meta-paths

From Figure 5, we can also analyze the impact of meta-paths. We have the following observations. (1) Compared with the methods



(a) *Recall*@100 (b) *MRR*@100 (c) *NDCG*@100

Figure 5: Performance of various GNNs on different graphs[3]

Table 3: Performance of different graph contrastive methods, trained with different objectives. Improve. computes performance divergence with respect to the *base* setting MGMP. Best performance under each objective category is underlined.

Objective	Model	Recall@100	MRR@100	NDCG@100
vanilla w/o counter-factual supplement	MGMP(base)	0.4981	0.1058	0.1838
	MGMP + SGL	0.4958	0.1063	0.1837
	MGMP + GCA	0.4967	0.1069	0.1842
	MGMP + simGCL	0.513	0.1111	0.1915
	MGMP + COSTA	0.5051	0.1079	0.1871
	MGMP + rand.	0.5173	0.1115	0.1926
	MGMP + DARP	0.5200	0.1125	0.1940
	Improve.	↑ 4.4 %	↑ 6.3 %	↑ 5.6 %
counter-factual supplement (supervised)	MGMP	0.5010	0.1079	0.1863
	MGMP + SGL	0.4965	0.1065	0.1840
	MGMP + GCA	0.4994	0.1119	0.1889
	MGMP + simGCL	0.5273	0.1138	0.1966
	MGMP + COSTA	0.5128	0.109	0.1895
	MGMP + rand.	0.5308	0.1153	0.1987
	MGMP + DARP	0.5331	0.1155	0.1991
	Improve.	↑ 7.0 %	↑ 9.2 %	↑ 8.3 %
counter-factual supplement (supervised +contrastive)	MGMP	0.5038	0.1092	0.1879
	MGMP + SGL	0.4981	0.1069	0.1846
	MGMP + GCA	0.5014	0.1082	0.1863
	MGMP + simGCL	0.5312	0.1107	0.1948
	MGMP + COSTA	0.5172	0.1105	0.1918
	MGMP + rand.	0.5424	0.1177	0.2028
	MGMP + DARP (**CC-GNN**)	**0.5450**	**0.1189**	**0.2046**
	Improve.	↑ 9.4 %	↑ 12.4 %	↑ 11.3 %

which do not use metapaths (i.e., GraphSAGE and AdaptiveGCN), the methods which use metapaths (i.e., LasGNN, MetaPath-guided Message Passing, and **CC-GNN**) can generally improve the search performance, regardless of the graph constructions. This pattern indicates that metapaths can better capture heterogeneous information on different graphs. (2) MGMP produces better results than LasGNN. The difference between MGMP and LasGNN is that MGMP samples two sub-graphs while LasGNN samples only one sub-graph. It proves that separating the click-path and phrase-path is beneficial. (3) The other components of **CC-GNN** can further improve the performance of MGMP. More discussions are provided in Section 4.5 and Section 4.6.

4.5 Ablation on Graph Contrastive Learning

To investigate the impacts of the proposed Difficulty-aware Representation Perturbation (DARP) in graph contrastive learning, we compare DARP with four SOTA graph contrastive learning methods that augment on graph structures or directly on representations. (1) SGL[4] contrasts the anchor graph with augmented (e.g., node dropout, edge dropout, etc.) graph [38]. (2) GCA[5] contrasts the anchor graph with graph augmented on both topology (i.e., structure dropout) and attribute levels (i.e., feature masking) [52]. (3) SimGCL[6] creates contrastive views by adding uniform noises to the embedding space at each graph convolutional layer [43]. (4) COSTA[7] uses a covariance-preserving feature space augmentation to construct a positive sample [47]. We also compare with a variant of DARP: (5) random perturbation (rand.) which adds a random noise at the query and item representation.

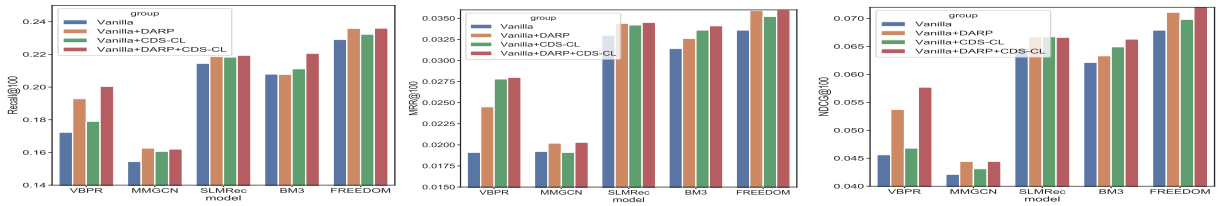
All methods are applied to the representation learned by MetaPath-guided Message Passing (MGMP). Since contrastive learning alone does not generate competitive search performances, the supervised learning loss is always incorporated. Furthermore, they are implemented to optimize three different objectives. (1) Vanilla supervised learning loss with true user click signals, but without counterfactual data supplement. (2) Supervised loss in Equation 4 which utilizes counterfactual supplement samples as "pseudo labels". (3) The ensemble loss in Equation 15 that incorporates counterfactual supplement at supervised and contrastive learning.

Results and Analysis. From Table 3, we have the following observations. (1) Augmenting the graph structure (i.e., SGL, GCA) is universally less effective than directly augmenting the feature representations (i.e., SimGCL, COSTA, rand. and DARP). In fact, the performances of SGL and GCA are worse than MGMP. This is probably because augmenting the graph structure is very likely to drop some key nodes and their associated edges. If the sub-graphs are broken into disconnected pieces because of the missing key nodes/edges, the original graph topology will be distorted. (2) *The proposed DARP performs best compared with other competitors and random perturbation regardless of the objectives.* The improvements are significant with respect to the base setting of MGMP. This proves that the learning difficulty is different for each node and that disturbance without considering learning difficulty is sub-optimal.

4.6 Ablation on Counterfactual Supplement

We analyze the results reported in Table 3 to investigate the performance gain by Counterfactual Data Supplement. (1) From Table 3, we observe positive impacts led by Counterfactual Data Supplement at both supervised and contrastive learning objectives. The results show that contrasting counterfactual positive and negative samples is a promising way to address the data scarcity problem in the E-commerce search. (2) Combining MGMP, DARP, and Counterfactual Data Supplement at both supervised and contrastive learning, we have **CC-GNN**, which produces the highest ranking performance.

[3]MGMP and **CC-GNN** are not implemented on BG because of the lack of phrase nodes in BG.
[4]https://github.com/wujcan/SGL-TensorFlow.
[5]https://github.com/CRIPAC-DIG/GCA.
[6]https://github.com/Coder-Yu/QRec.
[7]https://github.com/yifeiacc/COSTA.



(a) *Recall*@100 (b) *MRR*@100 (c) *NDCG*@100

Figure 6: Recommendation performance on Amazon Sports Dataset

Table 4: Performance of Counterfactual Data Supplement. The vanilla is MGMP + DARP.

Objective	Groups	Recall@100	MRR@100	NDCG@100
vanilla w/o	Head Query	0.5532	0.1212	0.2082
counterfactual	Tail Query	0.4416	0.0918	0.1699
supplement	Cold Item	0.4008	0.0793	0.1432
counterfactual	Head Query	0.5652	0.1235	0.2126
supplement	Tail Query	0.4579	0.0955	0.1664
(supervised)	Cold Item	0.4154	0.0820	0.1487
counterfactual	Head Query	**0.5778**	**0.1264**	**0.2179**
supplement	Tail Query	**0.4665**	**0.0991**	**0.1716**
(supervised +contrastive)	Cold Item	**0.4174**	**0.0831**	**0.1497**

(3) We point out here that *the performance gain by all components of **CC-GNN** is accumulative*. For example, we have already shown that the Content Collaborative Graph provides a better graph for various GNNs, and MetaPath-guided Message Passing outperforms other GNNs on Content Collaborative Graph. The Difficulty-aware Representation Perturbation improves MetaPath-guided Message Passing on Content Collaborative Graph, and combining Counterfactual Data Supplement at supervised and contrastive learning further improves the results obtained by Content Collaborative Graph. Every component of **CC-GNN** improves a step further than the previous components.

To investigate the impact of Counterfactual Data Supplement on different groups of queries, we again split the head queries, tail queries and cold start items following section 4.2. From Table 8, we observe that Counterfactual Data Supplement can improve the performance of different groups of queries and items simultaneously, alleviating the long-tail/cold-start problems.

4.7 Performance on Recommendation Models

CC-GNN consists of several components that can be stacked on other models and/or applied to other tasks. We have shown in Section A.3 that Content Collaborative Graph can improve the performance of different GNNs on product search problem. In this section, we want to investigate the proposed Difficulty-aware Representation Perturbation (DARP) and Counterfactual Data Supplement at Contrastive learning (CDS-CL). They can be easily plugged into a wide range of models, which is not restricted to graph learning methods. Furthermore, although the main purpose of **CC-GNN** is to solve industry-scale product search problem, DARP and CDS-CL can be applied to other tasks. In this section, we choose to study the multi-modal recommendation task, because of its wide usage [32, 37, 44, 50, 51].

The experimental proposal is: we apply DARP and/or CDS-CL on a backbone recommendation model. For a thorough study, we experiment with five representative backbones with different model architectures and learning paradigms. (1) VBPR [13] is a supervised, non-graph learning method which incorporates visual content features. (2) MMGCN [37] is a supervised, graph learning method which learns from modal-specific graph. (3) SLMRec [32] is a recent contrastive, graph learning method which proposes data augmentation on multi-modal content. (4) BM3 [51] is a recent contrastive, graph learning method on user-item interaction graph. (5) FREEDOM [50]) is a recent supervised, graph learning method which incorporates item-item graph with user-item graph.

For all backbones, we use the open source implementations [8]. We use Amazon sports dataset as mentioned in Section 4.1. Other implementation details are provided in the appendix.

Results and Analysis. From Figure 6, we observe that (1) Difficulty-aware Representation Perturbation (DARP) consistently increases the performance of different models. Specially, DARP achieves more significant improvements on supervised models (i.e., VBPR, MMGCN, and FREEDOM) than on contrastive learning models (i.e., SLMRec and BM3). One possible reason is that DARP is designed to differentiate item representations to distill additional supervisory signals. (2) Counterfactual Data Supplement at Contrastive Learning (CDS-CL) also consistently increases the performance of different models. Specially, CDS-CL obtains more improvements on contrastive learning methods (i.e., SLMRec and BM3). Contrastive learning focuses on the item itself and its augmentation. Long-tail/cold-start items are under-represented in the dataset and their representation learning benefit less from their augmentations. On the contrary, Counterfactual Data Supplement at Contrastive Learning is designed to transfer knowledge from head/warm items to long-tail/cold-start items, so it can further improve the performance of contrastive learning methods. (3) When combining DARP and CDS-CL, the best results can be obtained on all backbones. This phenomenon suggests that *DARP and CDS-CL can be stacked on various models and obtain maximal improvements from different perspectives.*

5 CONCLUSION

In this paper, we propose an efficient graph learning method **CC-GNN**, which shows superior performance on industry-scale product search, and alleviates the problems of long-tail queries and cold-start items. In addition, the proposed components of **CC-GNN**, such as the Content Collaborative Graph and Counterfactual Data Supplement, are shown to generally improve the performance of different GNN baselines on product search. Furthermore, experiments on public dataset show that applying the proposed components Difficulty-aware Representation Perturbation and Counterfactual Data Supplement at contrastive learning can improve the performance of different recommendation models. These results show that the proposed techniques have the potential to further enhance the performance of search and recommendation systems.

[8] https://github.com/enoche/MMRec

REFERENCES

[1] Ali Ahmadvand, Surya Kallumadi, Faizan Javed, and Eugene Agichtein. 2020. JointMap: Joint Query Intent Understanding For Modeling Intent Hierarchies in E-Commerce Search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. New York, NY, USA, 1509–1512. https://doi.org/10.1145/3397271.3401184

[2] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and Debias in Recommender System: A Survey and Future Directions. *CoRR* abs/2010.03240 (2020). arXiv:2010.03240

[3] Dian Cheng, Jiawei Chen, Wenjun Peng, Wenqin Ye, Fuyu Lv, Tao Zhuang, Xiaoyi Zeng, and Xiangnan He. 2022. IHGNN: Interactive Hypergraph Neural Network for Personalized Product Search. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. New York, NY, USA, 256–265. https://doi.org/10.1145/3485447.3511954

[4] Qiannan Cheng, Zhaochun Ren, Yujie Lin, Pengjie Ren, Zhumin Chen, Xiangyuan Liu, and Maarten de de Rijke. 2021. Long Short-Term Session Search: Joint Personalized Reranking and Next Query Prediction. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. New York, NY, USA, 239–248. https://doi.org/10.1145/3442381.3449941

[5] Arnon Dagan, Ido Guy, and Slava Novgorodov. 2021. An Image is Worth a Thousand Terms? Analysis of Visual E-Commerce Search. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. New York, NY, USA, 102–112. https://doi.org/10.1145/3404835.3462950

[6] Lu Fan, Qimai Li, Bo Liu, Xiao-Ming Wu, Xiaotong Zhang, Fuyu Lv, Guli Lin, Sen Li, Taiwei Jin, and Keping Yang. 2022. Modeling User Behavior with Graph Convolution for Personalized Product Search. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. New York, NY, USA, 203–212. https://doi.org/10.1145/3485447.3511949

[7] Shaohua Fan, Junxiong Zhu, Xiaotian Han, Chuan Shi, Linmei Hu, Biyu Ma, and Yongliang Li. 2019. Metapath-guided Heterogeneous Graph Neural Network for Intent Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*. ACM, 2478–2486. https://doi.org/10.1145/3292500.3330673

[8] Chenchen Feng, Yu He, Shiyang Wen, Guojun Liu, Liang Wang, Jian Xu, and Bo Zheng. 2022. DC-GNN: Decoupled Graph Neural Networks for Improving and Accelerating Large-Scale E-commerce Retrieval. In *Companion of The Web Conference 2022, Virtual Event / Lyon, France, April 25 - 29, 2022*. 32–40. https://doi.org/10.1145/3487553.3524203

[9] Alibaba group. 2019. LasGNN. Retrieved October 8, 2022 from https://github.com/alibaba/euler/wiki/LasGNN

[10] Parth Gupta, Tommaso Dreossi, Jan Bakus, Yu-Hsiang Lin, and Vamsi Salaka. 2020. Treating Cold Start in Product Search by Priors. In *Companion of The 2020 Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*. 77–78. https://doi.org/10.1145/3366424.3382705

[11] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 1024–1034.

[12] Ruining He and Julian J. McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*. 507–517. https://doi.org/10.1145/2872427.2883037

[13] Ruining He and Julian J. McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*. AAAI Press, 144–150.

[14] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong 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, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. 639–648. https://doi.org/10.1145/3397271.3401063

[15] Xiangnan He, Ming Gao, Min-Yen Kan, and Dingxian Wang. 2017. BiRank: Towards Ranking on Bipartite Graphs. *IEEE Trans. Knowl. Data Eng.* 29, 1 (2017), 57–71. https://doi.org/10.1109/TKDE.2016.2611584

[16] Bipul Kumar and Pradip Kumar Bala. 2017. Fattening The Long Tail Items in E-Commerce. *J. Theor. Appl. Electron. Commer. Res.* 12, 3 (2017), 27–49. https://doi.org/10.4067/s0718-18762017000300004

[17] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. 2018. Adaptive Graph Convolutional Neural Networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. 3546–3553.

[18] Sen Li, Fuyu Lv, Taiwei Jin, Guli Lin, Keping Yang, Xiaoyi Zeng, Xiao-Ming Wu, and Qianli Ma. 2021. Embedding-Based Product Retrieval in Taobao Search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Virtual Event, Singapore) *(KDD '21)*. New York, NY, USA, 3181–3189. https://doi.org/10.1145/3447548.3467101

[19] Jianghao Lin, Weiwen Liu, Xinyi Dai, Weinan Zhang, Shuai Li, Ruiming Tang, Xiuqiang He, Jianye Hao, and Yong Yu. 2021. A Graph-Enhanced Click Model for Web Search. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. 1259–1268. https://doi.org/10.1145/3404835.3462895

[20] Xinyi Liu, Wanxian Guan, Lianyun Li, Hui Li, Chen Lin, Xubin Li, Si Chen, Jian Xu, Hongbo Deng, and Bo Zheng. 2022. Pretraining Representations of Multi-modal Multi-query E-commerce Search. In *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*. 3429–3437. https://doi.org/10.1145/3534678.3539200

[21] Xiangyu Liu, Chuan Yu, Zhilin Zhang, Zhenzhe Zheng, Yu Rong, Hongtao Lv, Da Huo, Yiqing Wang, Dagui Chen, Jian Xu, Fan Wu, Guihai Chen, and Xiaoqiang Zhu. 2021. Neural Auction: End-to-End Learning of Auction Mechanisms for E-Commerce Advertising. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*. 3354–3364. https://doi.org/10.1145/3447548.3467103

[22] Hanqing Lu, Youna Hu, Tong Zhao, Tony Wu, Yiwei Song, and Bing Yin. 2021. Graph-based Multilingual Product Retrieval in E-Commerce Search. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*. Online, 146–153. https://doi.org/10.18653/v1/2021.naacl-industry.19

[23] Tharun Medini, Qixuan Huang, Yiqiu Wang, Vijai Mohan, and Anshumali Shrivastava. 2019. Extreme Classification in Log Memory using Count-Min Sketch: A Case Study of Amazon Search with 50M Products. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. 13244–13254.

[24] Erxue Min, Yu Rong, Tingyang Xu, Yatao Bian, Da Luo, Kangyi Lin, Junzhou Huang, Sophia Ananiadou, and Peilin Zhao. 2022. Neighbour Interaction based Click-Through Rate Prediction via Graph-masked Transformer. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*. 353–362. https://doi.org/10.1145/3477495.3532031

[25] Xichuan Niu, Bofang Li, Chenliang Li, Rong Xiao, Haochuan Sun, Honggang Wang, Hongbo Deng, and Zhenzhong Chen. 2020. Gated Heterogeneous Graph Representation Learning for Shop Search in E-Commerce. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management* (Virtual Event, Ireland) *(CIKM '20)*. New York, NY, USA, 2165–2168. https://doi.org/10.1145/3340531.3412087

[26] Jérémie Rappaz, Julian J. McAuley, and Karl Aberer. 2021. Recommendation on Live-Streaming Platforms: Dynamic Availability and Repeat Consumption. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September 2021 - 1 October 2021*. 390–399. https://doi.org/10.1145/3460231.3474267

[27] Israr Ur Rehman, Waqar Ali, Zahoor Jan, Zulfiqar Ali, Hui Xu, and Jie Shao. 2023. CAML: Contextual augmented meta-learning for cold-start recommendation. *Neurocomputing* 533 (2023), 178–190. https://doi.org/10.1016/j.neucom.2023.02.051

[28] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The Graph Neural Network Model. *IEEE Trans. Neural Networks* 20, 1 (2009), 61–80. https://doi.org/10.1109/TNN.2008.2005605

[29] Walid Shalaby, Sejoon Oh, Amir Afsharinejad, Srijan Kumar, and Xiquan Cui. 2022. M2TRec: Metadata-aware Multi-task Transformer for Large-scale and Cold-start free Session-based Recommendations. In *RecSys '22: Sixteenth ACM Conference on Recommender Systems, Seattle, WA, USA, September 18 - 23, 2022*. ACM, 573–578. https://doi.org/10.1145/3523227.3551477

[30] Daria Sorokina and Erick Cantú-Paz. 2016. Amazon Search: The Joy of Ranking Products. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*. 459–460. https://doi.org/10.1145/2911451.2926725

[31] Theban Stanley, Nihar Vanjara, Yanxin Pan, Ekaterina Pirogova, Swagata Chakraborty, and Abon Chaudhuri. 2020. SIR: Similar Image Retrieval for Product Search in E-Commerce. In *Similarity Search and Applications - 13th International Conference, SISAP 2020, Copenhagen, Denmark, September 30 - October 2, 2020, Proceedings (Lecture Notes in Computer Science, Vol. 12440)*. 338–351. https://doi.org/10.1007/978-3-030-60936-8_26

[32] Zhulin Tao, Xiaohao Liu, Yewei Xia, Xiang Wang, Lifang Yang, Xianglin Huang, and Tat-Seng Chua. 2022. Self-supervised Learning for Multimedia Recommendation. *IEEE Transactions on Multimedia* (2022), 1–10. https://doi.org/10.1109/TMM.2022.3187556

[33] Dai Hoang Tran, Abdulwahab Aljubairy, Munazza Zaib, Quan Z. Sheng, Wei Emma Zhang, Nguyen H. Tran, and Khoa L. D. Nguyen. 2020. HeteGraph: A Convolutional Framework for Graph Learning in Recommender Systems. In *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*. 1–8. https://doi.org/10.1109/IJCNN48605.2020.9207078

[34] Dai Hoang Tran, Quan.Z Sheng, Wei Emma Zhang, Abdulwahab Aljubairy, Munazza Zaib, Salma Abdalla Hamad, Nguyen H. Tran, and Nguyen Lu Dang Khoa.

2021. HeteGraph: graph learning in recommender systems via graph convolutional networks. *Neural Computing and Applications* (2021), 1–17.

[35] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

[36] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*. 165–174. https://doi.org/10.1145/3331184.3331267

[37] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. 2019. MMGCN: Multi-modal Graph Convolution Network for Personalized Recommendation of Micro-video. In *Proceedings of the 27th ACM International Conference on Multimedia, MM 2019, Nice, France, October 21-25, 2019*. 1437–1445. https://doi.org/10.1145/3343031.3351034

[38] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised Graph Learning for Recommendation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. 726–735. https://doi.org/10.1145/3404835.3462862

[39] Tao Wu, Ellie Ka In Chio, Heng-Tze Cheng, Yu Du, Steffen Rendle, Dima Kuzmin, Ritesh Agarwal, Li Zhang, John R. Anderson, Sarvjeet Singh, Tushar Chandra, Ed H. Chi, Wen Li, Ankit Kumar, Xiang Ma, Alex Soares, Nitin Jindal, and Pei Cao. 2020. Zero-Shot Heterogeneous Transfer Learning from Recommender Systems to Cold-Start Search Retrieval. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*. 2821–2828. https://doi.org/10.1145/3340531.3412752

[40] Yiqing Wu, Ruobing Xie, Yongchun Zhu, Xiang Ao, Xin Chen, Xu Zhang, Fuzhen Zhuang, Leyu Lin, and Qing He. 2022. Multi-view Multi-behavior Contrastive Learning in Recommendation. In *Database Systems for Advanced Applications - 27th International Conference, DASFAA 2022, Virtual Event, April 11-14, 2022, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 13246)*. 166–182. https://doi.org/10.1007/978-3-031-00126-0_11

[41] Ruobing Xie, Qi Liu, Liangdong Wang, Shukai Liu, Bo Zhang, and Leyu Lin. 2022. Contrastive Cross-domain Recommendation in Matching. In *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*. 4226–4236. https://doi.org/10.1145/3534678.3539125

[42] Fan Yang, Ajinkya Kale, Yury Bubnov, Leon Stein, Qiaosong Wang, M. Hadi Kiapour, and Robinson Piramuthu. 2017. Visual Search at eBay. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*. 2101–2110. https://doi.org/10.1145/3097983.3098162

[43] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are Graph Augmentations Necessary?: Simple Graph Contrastive Learning for Recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*. 1294–1303. https://doi.org/10.1145/3477495.3531937

[44] Jinghao Zhang, Yanqiao Zhu, Qiang Liu, Shu Wu, Shuhui Wang, and Liang Wang. 2021. Mining Latent Structures for Multimedia Recommendation. In *MM '21: ACM Multimedia Conference, Virtual Event, China, October 20 - 24, 2021*. 3872–3880. https://doi.org/10.1145/3474085.3475259

[45] Yanhao Zhang, Pan Pan, Yun Zheng, Kang Zhao, Yingya Zhang, Xiaofeng Ren, and Rong Jin. 2018. Visual Search at Alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2018, London, UK, August 19-23, 2018*. 993–1001. https://doi.org/10.1145/3219819.3219820

[46] Yuan Zhang, Dong Wang, and Yan Zhang. 2019. Neural IR Meets Graph Embedding: A Ranking Model for Product Search. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*. 2390–2400. https://doi.org/10.1145/3308558.3313468

[47] Yifei Zhang, Hao Zhu, Zixing Song, Piotr Koniusz, and Irwin King. 2022. COSTA: Covariance-Preserving Feature Augmentation for Graph Contrastive Learning. In *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*. 2524–2534. https://doi.org/10.1145/3534678.3539425

[48] Kai Zhao, Yukun Zheng, Tao Zhuang, Xiang Li, and Xiaoyi Zeng. 2022. Joint Learning of E-Commerce Search and Recommendation with a Unified Graph Neural Network. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining* (Virtual Event, AZ, USA) *(WSDM '22)*. New York, NY, USA, 1461–1469. https://doi.org/10.1145/3488560.3498414

[49] Yujia Zheng, Siyi Liu, Zekun Li, and Shu Wu. 2021. Cold-start Sequential Recommendation via Meta Learner. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 4706–4713.

[50] Xin Zhou. 2022. A Tale of Two Graphs: Freezing and Denoising Graph Structures for Multimodal Recommendation. *CoRR* abs/2211.06924 (2022). https://doi.org/10.48550/arXiv.2211.06924 arXiv:2211.06924

[51] Xin Zhou, Hongyu Zhou, Yong Liu, Zhiwei Zeng, Chunyan Miao, Pengwei Wang, Yuan You, and Feijun Jiang. 2023. Bootstrap Latent Representations for Multimodal Recommendation. In *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*. ACM, 845–854. https://doi.org/10.1145/3543507.3583251

[52] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph Contrastive Learning with Adaptive Augmentation. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. New York, NY, USA, 2069–2080. https://doi.org/10.1145/3442381.3449802

A APPENDIX

A.1 Implementation

A.1.1 Experiments on IPQS Dataset. The parameters are initialized with Glorot initialization and trained using Adam optimizer.

On IPQS dataset, for all baselines, competitors, and **CC-GNN** variants, the input title features are 50-dimensional vectors extracted from a word2vec model trained on an in-house large-scale E-commerce corpus; the input image features are 512-dimensional vectors extracted from a metric learning model trained on an E-commerce visual search dataset. Sparse features include continuous features and discrete features. For continuous features like prices and sales, we segment them into intervals and convert each interval into a one-hot feature. For other discrete features, such as cities and ids, we use one-hot features directly. The size of the three types of embeddings D is 128. They are updated simultaneously during training. Item embeddings $\mathbf{t}$ and query embeddings $\mathbf{q}$ are updated according to the Loss function in Equation 15. Phrase embeddings $\mathbf{p}$ are updated through the phrase meta-path in the graph propagation process.

The product titles are in Chinese, and word segmentation and named entity recognition are implemented using an open-source tool [9].

Except in Appendix A.3 and Appendix A.4, the score threshold $\xi_s = 6$, and the content similarity threshold $\xi = 0.85$. We use visual similarity as content similarity. We use 2-layer GAT as the aggregator. The number of negative samples N is 5. The hyper-parameters $\alpha = 0.6$, $\beta = 0.2$.

Table 5: Manually defined entities type scoring table

Entity	Score	Entity	Score
Category	1.0	Quality	1.0
Category modifier	1.0	Proper noun	1.0
Brand	1.0	Prefix	0.75
Material	1.0	Suffix	0.75
Color	1.0	Unit	0.5
Efficacy	1.0	Time and Season	0.5
People	1.0	Symbol	0.25
Size	1.0	Number	0.25
Location	1.0	Common	0.1

A.1.2 Named entity recognition tool and manually defined entities type scoring table. We utilize an open-source E-commerce Named Entity Recognition tool [10] developed by Alibaba Group, the largest e-commerce platform in China. This Named Entity Recognition (NER) tool is specifically designed for e-commerce scenarios and segments each e-commerce entity into 18 corresponding entity types such as Category, Material, Color, Common, and so on. It also provides weights for each entity type (i.e., the scores in Table 5), which are based on extensive e-commerce data experience.

[9] https://m.nlp.aliyun.com/

[10] https://help.aliyun.com/document_detail/200996.html?spm=a2c4g.392277.0.0.5bb6a4baPBFzeG

A.1.3 Reordering the words in a phrase alphabetically. We mainly focus on incorporating content collaboration. Phrases with the same words but different word orders usually have equivalent or similar roles in content collaboration. If we treat them as distinct nodes, the collaboration between the item and the query nodes is dispersed, hindering the process of integrating content collaboration. It will also significantly increase the computational cost in constructing the graph and storage cost in large-scale industry graphs. Thus, we rewrite each phrase and reorder the words in a phrase in alphabetical order. For example, "fine wine glasses" and "wine glasses fine" will be rewritten as "fine glasses wine".

A.1.4 Experiments on Amazon Sports Dataset. The textual feature dimension is 384 and the visual feature dimension is 4,096. The size of user and item embeddings is 64. We use Adam as the optimizer with a learning rate of 0.001. For convergence consideration, the early stopping and total epochs are fixed at 10 and 200, respectively.

A.2 The impact of adding content nodes

Most previous studies add nodes that are already existing in the system [3, 6, 7, 18]. For example, MEIRec [7] adds query nodes to a user-item graph to represent the intent in the task of intent recommendation. The query nodes already exist in the system and have explicit interactions with other entities in the graph. In contrast, we add virtual entities that are extracted from product contents to provide auxiliary information. In other words, the added nodes did not exist separately and did not have an identity of their own in the system.

We argue that these virtual entities from product content play an important role in information propagation. For example, a user wants to buy a TV. He/she enters a query, and his/her purchase is affected by the catchy phrase "impressive contrast of mini-LED" in an ad. The other TV makers notice the trending phrase and start to include this phrase in their product content. Thus, the phrase "impressive contrast of mini-LED" implicatively participates in the information propagation and is not present in the original query-item graph.

We would like to point out that adding content information to the original graph is non-trivial. For example, to our knowledge, there is one recent work in cross-domain recommendation [41] that adds word nodes (i.e., a node represents a word and connects to the item nodes which use this word to describe them). However, this approach has two major drawbacks in our scenario: (1) Firstly, there will be a large number of extra links connecting to the word nodes, which results in significant storage and computational costs; (2) Secondly, word nodes that do not provide complete and clear semantic information can hinder information propagation. Therefore, in our work, we extract phrases from product titles and queries and then prune phrases that are too vague and irrelevant to any product feature.

To investigate the performance of different graph constructions, we conduct experiments on the IPQS dataset. We have four different graph constructions, including (1) original bipartite query-item graph (BG), (2) adding word nodes to the original graph (WG), (3) adding phrase nodes to the original graph (Tripartite query-item-phrase graph, TG), and (4) adding pruned phrase nodes to the original graph (Content Collaborative Graph, CCG). We use LasGNN as the backbone and report the performance regarding

Table 8: Performance of Counterfactual Data Supplement at Contrastive Learning with different ξ_c.

Vanilla	Groups	Recall@100	MRR@100	NDCG@100
BM3	vanilla	0.2079	0.0314	0.0621
	$\xi_c = 0$	0.2041	0.0298	0.0610
	$\xi_c = 0.2$	0.2106	0.0309	0.0622
	$\xi_c = 0.4$	0.2102	0.0311	0.0623
	$\xi_c = 0.6$	0.2110	0.0317	0.0631
	$\xi_c = 0.8$	**0.2131**	**0.0334**	**0.0648**
	$\xi_c = 1$	0.2122	0.0313	0.0629
FREEDOM	vanilla	0.2290	0.0336	0.0679
	$\xi_c = 0$	0.2264	0.0333	0.0671
	$\xi_c = 0.2$	0.2262	0.0337	0.0673
	$\xi_c = 0.4$	0.2257	0.0335	0.0671
	$\xi_c = 0.6$	0.2257	0.0338	0.0674
	$\xi_c = 0.8$	**0.2415**	**0.0363**	**0.0726**
	$\xi_c = 1$	0.2354	0.0352	0.0704

Table 6: Performance of LasGNN with different graphs

Graph	Recall@100	MRR@100	NDCG@100	#Nodes	#Edges
BG	0.4756	0.0986	0.1735	0	0
WG	0.3410	0.0611	0.1141	4.0 million	1,220 million
TG	0.4691	0.0935	0.1678	16.4 million	127 million
CCG ($\xi_s = 6$)	**0.4820**	**0.1058**	**0.1831**	6.0 million	24 million

Table 7: Performance of MGMP with different graphs

Graph	Recall@100	MRR@100	NDCG@100	#Phrases
BG	0.4805	0.1038	0.1759	0
TG ($\xi_s = 0$)	0.4763	0.0981	0.1734	16.4 million
CCG ($\xi_s = 2$)	0.4832	0.1015	0.1753	14.7 million
CCG ($\xi_s = 4$)	0.4887	0.1056	0.1782	12.9 million
CCG ($\xi_s = 6$)	**0.4981**	**0.1079**	**0.1838**	6.0 million
CCG ($\xi_s = 8$)	0.4942	0.1066	0.1819	2.9 million

Recall@100, *MRR*@100, and *NDCG*@100. We also report the number of nodes (#Node) and the number of edges (#Edges) added to the original graph.

Results and Analysis. We have the following observations: (1) The number of edges greatly increases on WG and BG, but the performance on WG and TG is worse than BG. This shows that *adding auxiliary information to the original graph is difficult.* Words and unpruned phrases significantly increase the storage and computational cost, but do not provide complete and clear semantic information and hinder information propagation. (2) CCG outperforms BG. This shows that *adding auxiliary information via virtual phrase nodes is effective.*

A.3 The impact of phrase pruning on CC-GNN

The threshold ξ_s in Section 3.2.2 controls the number of phrases to be pruned. Thus we experiment with different values of ξ_s. For example, as shown in Table 7, TG can be considered as CCG with $\xi_s = 0$. When $\xi_s = 6$ CCG removes 10.4 million phrase (7.5%) nodes and 103 million edges (12.3%) in TG. We run the basic building block of **CC-GNN**, i.e., MetaPath-guided Message Passing (MGMP), on the different graphs. Note that running MGMP on BG is equivalent to running LasGNN on BG, because BG does not contain phrase paths.

Results and Analysis. We have the following observations from Table 7. (1) *Phrase pruning are important.* On one hand, the performance of WG and TG is worse than BG. This implies that simply including phrases in the graph is not helpful. Since the threshold ξ_s is positively correlated to the length of phrases, this result also indicates that *short phrases have insufficient information and they occupy many computing and storage resources.* On the other hand, when the threshold $\xi_s < 6$, increasing threshold ξ_s significantly reduces the number of phrases in the graph and gradually improves *Recall*@100, *MRR*@100 and *NDCG*@100. This trend suggests that phrase pruning can increase both storage efficiency and search effectiveness. (2) *The best performance is achieved on* $\xi_s = 6$, when MGMP achieves the best results while reducing 7.5% nodes comparing to TG. (3) When the threshold $\xi_s = 8$, the number of phrases is greatly reduced, but the performance slightly decreases. The underlying reason is a high threshold will incorrectly prune some meaningful short phrases.

A.4 The impact of content threshold ξ_c

The content threshold ξ_c controls the content similarity in Counterfactual Data Supplement (CDS). $\xi_c in [0, 1]$ suggests that only the item content similarity higher than ξ_c will be selected as a positive sample in CDS. To investigate the impact of ξ_c, we implement a recommendation task on the Amazon Sports Dataset. We apply CDS-CL with different values of ξ_c on the BM3 and FREEDOM models and report the recommendation performance regarding *Recall*@100, *MRR*@100, and *NDCG*@100. Note that the vanilla version is the original BM3 and FREEDOM.

Results and Analysis. We have the following observations: (1) An intermediate value of ξ_c can significantly increase the performance on both models. $\xi_c = 0.8$ achieves the best performance, and the *Recall*@100, *MRR*@100, and *NDCG*@100 are increased by 2.5%, 6.4%, 4.3% on BM3, 5.5%, 8.0%, 6.9% on FREEDOM. (2) For $\xi_c < 0.8$, smaller ξ_c leads to worse performance. This phenomenon suggests that head and tail items with a low content similarity should not be considered as positive samples in CDS, which verifies the necessity of using the content similarity threshold ξ_c. (3) The performance drops at $\xi_c = 1$. The reason is that we only utilize a small number of item pairs with identical content and miss the opportunities to further enhance the performance with less similar tail/cold items. However, as CDS-CL alone can enhance the content representation learning and eliminate the prediction dependence on click numbers, the performance is still better than the vanilla models.