Enhancing Catalog Relationship Problems with Heterogeneous Graphs and Graph Neural Networks Distillation

Boxin Du[1], Rob Barton[1], Grant Galloway[1], Junzhou Huang[1,2], Shioulin Sam[1], Ismail Tutar[1], Changhe Yuan[1]
[1]Amazon
[2]University of Texas at Arlington, USA

ABSTRACT

Traditionally, catalog relationship problems in e-commerce stores have been handled as pairwise classification tasks, which limit the ability of machine learning models to learn from the diverse relationships among different entities in the catalog. In this paper, we leverage heterogeneous graphs and Graph Neural Networks (GNNs) for improving catalog relationship inference. We start from investigating how to create multi-entity, multi-relationship graphs from diverse relationship data sources, and then explore how to utilizing GNNs to leverage the knowledge of the constructed graph in a self-supervised fashion. We finally propose a distillation approach to transfer the knowledge learned by GNNs into a pairwise neural network for seamless deployment in the catalog pipeline that relies on pairwise input for inductive relationship inference. Our experiments exhibit that in two of the representative catalog relationship problems, Title Authority/Contributor Authority and Broken Variation, the proposed framework is able to improve the recall at 95% precision of a pairwise baseline by up to 33.6% and 14.0%, respectively. Our findings highlight the effectiveness of this approach in advancing catalog quality maintenance and accurate relationship modeling, with potential for broader industry adoption.

ACM Reference Format:
Boxin Du[1], Rob Barton[1], Grant Galloway[1], Junzhou Huang[1,2], Shioulin Sam[1], Ismail Tutar[1], Changhe Yuan[1]. 2023. Enhancing Catalog Relationship Problems with Heterogeneous Graphs and Graph Neural Networks Distillation. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The catalog of a large e-commerce store such as Amazon and eBay is a vast database of structured products, each defined by various attributes such as product types, contributors, browse nodes, and more. High-quality item relationships are critical to ensuring a positive online shopping experience for customers. For instance, grouping similar items in search results helps customers find the products they want without feeling overwhelmed by a large number of disorganized items. All the closely matched items fall into a relationship category where they may be functionally the same but differ in certain attributes. Due to the constraints of the traditional architecture of the catalog pipeline, the approach to many catalog relationship problems relies on pairwise machine learning models which learn from item pairs. They usually have limitations in learning from other item entities and relationships. Graphs, on the other hand, are a ubiquitous and powerful tool for representing relationships between entities in various domains. By leveraging graphs that connect different types of entities and rich semantic information, it is possible to overcome the limitation of pairwise models and enhance their performance. An illustrative example is shown in Figure 2. Despite the potential benefits, the utilization of graphs and graph mining methods in catalog relationship problems is currently underexplored.

Existing research on utilizing graphs for catalog relationship problems has mainly focused on individual relationships. However, different entities and their underlying relationships are interconnected, and their combination could be beneficial to individual problems. For example, a book item could be a member of both a title set, which clusters books of the same title, and it can also be connected to a browse node (BN) in BN hierarchy, which is a hierarchical taxonomy for organizing items often used in e-commerce stores. Books from the same title set are highly likely to be linked to the same BN. Creating a heterogeneous graph to link different types of entities could potentially help to learn from multiple entities and their relationships. Recent advances in Graph Neural Networks (GNNs) also provide a powerfully tool in solving a variety of machine learning problems on graphs [5, 6, 13, 20, 21]. However, there are several key challenges for practitioners when applying GNNs to traditional catalog relationship problems. First, when creating a multi-entity, multi-relationship graph, it is difficult to explain the functionality of the connections between entities, and practitioners often overlook which types of relationships would be noisy to the downstream applications. Second, human labeled data in a constructed graph is often sparse. How to leverage the knowledge of the graph in a self-supervised fashion when audited data is limited? Third, GNNs require graphs instead of individual pairs for inference, but in the relationship inference, the target nodes/edges may not be connected to the training graph. Such inductive setting should be mandatory in the catalog system, since there will be constantly new incoming items to the catalog. Fourth, the difference in the model interface between the traditional pairwise models and GNNs would make drastic shift in model deployment logic. Furthermore, performing GNN inference on massive graphs is computationally challenging. How to transfer the knowledge learned by GNNs to an industry-scale catalog system through efficient deployment?

To address these challenges, we propose a framework to utilize heterogeneous graphs for the catalog relationship problems.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

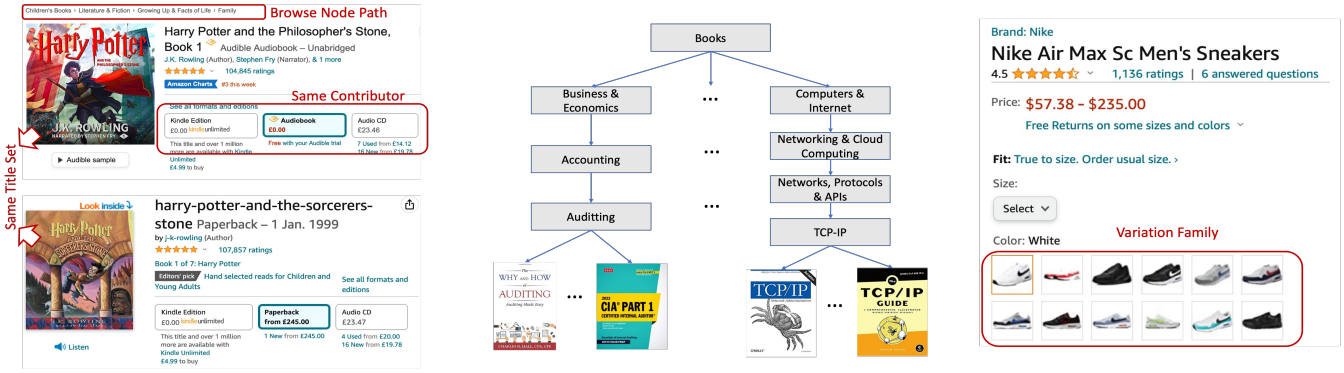

Figure 1: An examples of TA/CA relationship (a), Browse Node path (a), Browse Node (BN) hierarchy (b), and Variation Family (c). In (a), the two buyable book items are in different editions, but belong to the same title set. Different formats of the same item (e.g., Kindle, Audiobook, etc.) share one contributor, J.K. Rowling. (b) shows a BN tree rooted at books. Multiple items could be assigned to each leaf node. (c) shows a Variation Family of Nike shoes in a detail page.

The framework includes creating a multi-entity, multi-relationship heterogeneous graph from different data sources, and then adopting GNNs for on-graph self-supervised and supervised learning. Finally, the knowledge of a GNN model is distilled to a pairwise neural network for supporting efficient deployment and inductive inference in the catalog pipeline that relies on pairwise input. We present experiments on two representative problems in catalog relationship, Title Authority/Contributor Authority (TA/CA) and Broken Variation (BV), but we places our focus on the TA/CA use case, as the construction of the graph requires a case-by-case study. Our contributions are:

- We propose a general strategy of creating multi-entity, multi-relationship graph for a target catalog relationship problem, and analyze its optimality in the experiments.
- We explore how to pre-train on the constructed heterogeneous graph for leveraging massive, unlabeled graph data.
- We propose a distillation method to transfer the heterogeneous graph knowledge learned from a GNN model to a pairwise model for inductive inference. We show that the proposed method is generic, and can be applied to various catalog relationship problems for broader impact.
- We conduct extensive experiments, and show that the distilled GNN model is able to significantly outperform the baseline models across multiple catalog problems.

2 PRELIMINARIES

A - TA/CA Relationship. The TA and CA catalog relationships group together media products such as books, music, or videos based on their primary content or contributor, respectively. For TA, each child item represents a buyable product visible to customers, while the parent item represents the title set and is non-buyable and invisible to customers. The title set could be, for example, the same book differing across binding (paperback, hardback, kindle, etc.) or a film across DVD, BlueRay, Prime Video formats. An example can be seen in Figure 1 (a). The example books from different marketplaces and different editions belong to the same title set.

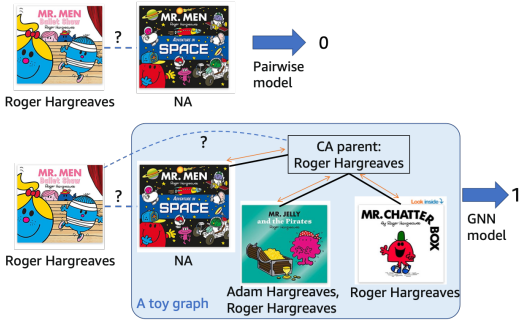

Figure 2: An example of predicting if a pair of items share the same contributor in Contributor Authority problem (CA). When there are missing attributes (e.g., contributor name), the pairwise model often assigns low probabilities and gives negative prediction. Using a graph would help to learn embeddings of items from its siblings and parents.

CA is another parent-child catalog relationship that groups together media products with the same contributor (e.g. author, editor, artist etc.). Parent items (non-buyable) represent contributor entities with many child items (buyable) representing products.

B - Browse Node Relationship. As shown in Figure 1 (b), a browse tree is a hierarchical taxonomy for organizing items often used in e-commerce stores. A browse node is a named location in a browse tree that is used for navigation, product classification, and website content. Browse nodes are attributes visible on the e-commerce webpage that help customers navigate through the vast selection of products and find the item's they are looking for or discover new items to buy.

C - Variation and Broken Variation. A Variation refers to a group of products that share similar attributes but differ in specific features such as size, color, quantity, and so on. By establishing a variation relationship between products, they can be displayed on a single detail page, allowing customers to choose from the available options without navigating to different web pages.

However, when families of products are created, mistakes may occur due to human error or tool malfunction, resulting in items having various errors. This can prevent the variation family from displaying correctly on the website. A Broken Variation is a term used to highlight multiple sets of items that should belong to a single family but are somehow split into several. The goal of the BV task is to correct the affected items.

D - Pairwise Baseline Model. Many existing systems in this field are based on pairwise machine learning models which predict whether a relationship exists in pair of input items. For TA/CA problem, the pairwise model is usually a Siamese Net [1]. The input item pair are first fed into a parameter-sharing embedding layer. Then the output item embeddings are merged, and concatenated with other available pairwise features as the input of a feed-forward neural network (FFN). The output of the FFN is the classification prediction. An illustration can be seen in the right side of Figure 4.

3 THE PROPOSED FRAMEWORK

Overall, our framework converts relationship prediction on item pairs to edge classification problem using GNNs on a multi-entity, multi-relationship heterogeneous graph, which could be either constructed from the data of a single problem, or multiple problems. Then the GNN model is distilled by a pairwise neural network for inductive inference.

3.1 Graph Creation

Theoretically, when creating a heterogeneous graph around a set of items, one can include as many types of entities and relationships as possible, as long as they can be connected to the items in a reasonable way. However, including too many types of nodes/edges may introduce noise, redundancy, and overly sparse or dense connections. Therefore, our graph creation approach follows a *single task -> multi-task -> de-noising* procedure, which consists of four steps. Firstly, a set of seed items is sampled from a specific target scope (e.g., a set of books items from a marketplace). Secondly, relevant catalog entities of the seed items in the target scope and the audited dataset of a specific catalog relationship task are merged to create a labeled graph, where nodes and edges are created for the entities and their corresponding relationships. The output graph is named the Single-task Graph. Third, multiple Single-task Graphs are combined to create a Multi-task Graph for closely related tasks, which can result in a heterogeneous graph with multiple types of entities and relationships. However, this graph may not be optimal for the target task, so the last step is to prune it by removing nodes/edges with negative impact to downstream tasks.

Based on this strategy, in TA/CA problem, we first create a Single-task TA/CA Graph consisting of TA/CA parents and items as nodes, and three types of edges: (i) the edges between TA/CA parents and items that belong to TA/CA parents to represent TA/CA membership; (ii) the edges between TA and CA parents for `TA_belongTo_CA` relationship; and (iii) the edges between items from audited item pairs. The audited item pairs of TA/CA labels are accumulated for training the existing pairwise baseline models over the years, and they are merged with the sampled TA/CA clusters from the catalog. We further construct a Multi-task Graph illustrated in Figure 3. This heterogeneous graph consists of three Single-task Graphs from

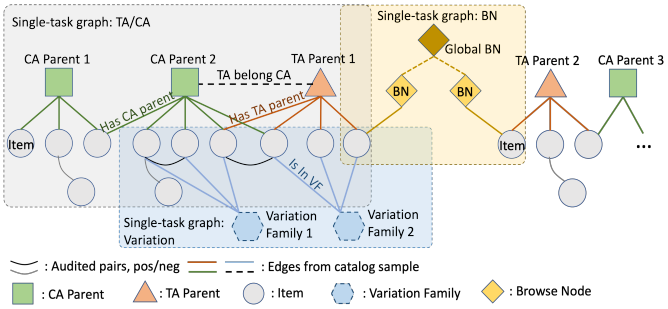

Figure 3: The multi-entity and multi-relationship heterogeneous graph of three catalog relationships.

three catalog relationships, TA/CA, BN and Variation. However, the Variation Graph has very sparse links from the variation family nodes to the items of the TA/CA graph, because variations are not common within the media scope of TA/CA relationship. Therefore they are safely removed and not utilized in the experiments. We will detail the analysis of the optimality and pruning of the Multi-task Graph in Section 4. Similarly, for BV problem, we create a Single-task BV Graph from both the sampled Variation clusters from the catalog and audited data in the experiments.

3.2 Model Architecture and Training

After graph construction, we conduct on-graph learning for TA/CA problem, followed by off-graph distillation. First, we adopt RGCN model [19] for message passing and aggregation of node embeddings on the created graph, for its ability of modeling different edge relationships. We initialize the node embeddings of items with pre-trained embeddings from text attributes, and the embeddings of TA/CA and BN with one-hot encodings. We also generate edge features for edges between two items from a variety of item attributes, including text, numerical, and categorical types. We adjust the standard RGCN layer [19] to leverage edge features in message passing as follows.

$$\mathbf{h}_i^{(l+1)} = \sigma(\mathbf{W}_0^{(l)}\mathbf{h}_i^{(l)} + \sum_{r\in R}\sum_{j\in N_i^r}\frac{1}{c_{i,r}}\mathbf{W}_r^{(l)}\mathbf{h}_j^{(l)} + \sum_{r\in R}\sum_{j\in N_i^r}\frac{1}{d_{i,r}}\tilde{\mathbf{W}}_r^{(l)}\mathbf{p}_{i,j}) \tag{1}$$

In Eq. (1), $\tilde{\mathbf{W}}_r^{(l)}$ is a learnable weight matrix for message passing of edges which belong to relationship r. $\mathbf{p}_{i,j}$ is the pairwise feature between node (i, j), and $d_{i,r}$ is a normalization constant.

In a large heterogeneous graph such as the Multi-task Graph we create in Figure 3, the label information is in general quite limited. For example, in the Multi-task Graph consisting of the sampled TA/CA and BN data used in the experiment, only 68K edges have labels out of over 4MM edges in total. In order to fully utilize the graph structure information when using GNNs, we propose a self-supervised approach to pre-train the GNN model on the massive data of the unlabeled graph, before finetune the GNN model on the labeled edges. Specifically, the goal of the TA/CA problem on graph is to predict the class of the labeled `<item, share_TA/CA_parent, item>` edges, so we try to apply link prediction on the unlabeled `<item, has_TA/CA_parent, TA/CA parent>` edges for pre-training task, since they are the most related to the supervised task.

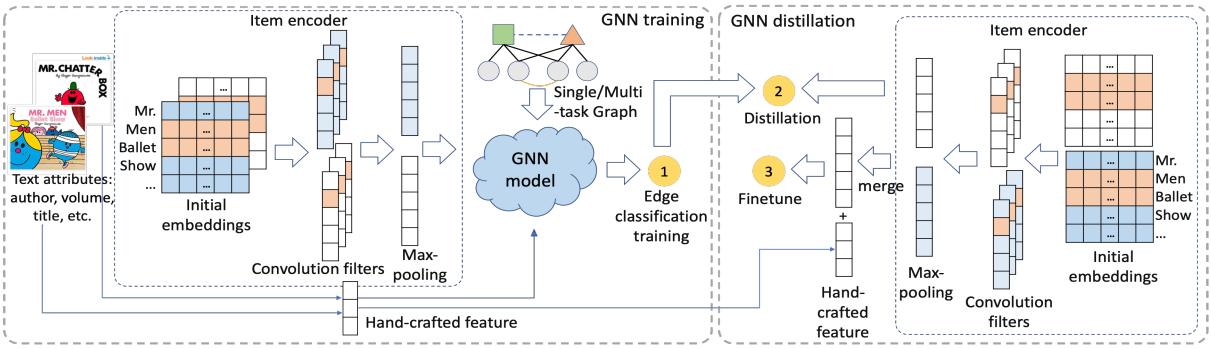

Figure 4: The pipeline of RGCN model training and distillation for TA/CA.

During link prediction, for an existing target <item, has_TA/CA_parent, TA/CA parent> edge, we uniformly sample k negative node pairs which have the same source and target node type from the graph, but no edges exist between them. We exclude the target <item, has_TA/CA_parent, TA/CA parent> edge during message passing of the RGCN model. The loss function for link prediction in pre-training is as follows.

$$\mathcal{L}^{pre} = \sum_{i,j}[log(1 - \text{FFN}(\mathbf{h}_i^{(s)}||\mathbf{h}_i^{(d)})) + log(\text{FFN}(\mathbf{h}_i^{(s)}||\mathbf{h}_j^{(n)}))] + \alpha(\sum_{r,l}||\mathbf{W}_r^{(l)}|| + ||\mathbf{W}_0^{(l)}||) \quad (2)$$

where $\mathbf{h}_i^{(s)}$ and $\mathbf{h}_i^{(d)}$ are the node embeddings from the RGCN model of the source and destination node of a positive edge i. $\mathbf{h}_j^{(n)}$ is the j-th negative node embedding corresponding to the source node in edge i. The positive node embedding pairs and negative node embeddings pairs are concatenated and fed into a Feed-Forward Neural Network (FFN) for the prediction logits. The second term of Eq. (2) is a regularization of the dense parameters in the RGCN.

After pre-training, the labeled <item, share_TA_parent, item> edges are used for finetuning with edge classification task. During edge classification, the node embeddings are initialized by the node embeddings from pre-training. A binary cross-entropy loss is adopted for the edge classification finetuing.

Besides using link prediction for pre-training, there are also other possible self-supervised pretext tasks suitable for learning the graph structure, such as edge classification. For example, we can sample item pairs of the same TA/CA parents as positive edges, and item pairs from different TA/CA parents as negative edges. But the advantage of link prediction over edge classification is that the edges for link prediction are readily available, while one needs to sample positive and negative edges for edge classification. Furthermore, the sampling space for item pairs grows exponential w.r.t. the number of items, which makes it difficult to decide the ideal size of the samples.

3.3 GNN Model Distillation

After RGCN training is finished, we distill the RGCN model by a Siamese Net that shares the same architecture as the baseline model (right side of Figure 4). The specific distillation process differs slightly in TA/CA and BV problems, due to the task and graph differences. Generally, we can distill the knowledge of a well-trained GNN model by using either node/edge embeddings or node/edge soft labels as targets. For TA/CA problem, we first use the trained RGCN model to infer the item embeddings for all the item nodes in the TA/CA graph. Then we use these item embeddings as targets to train the item encoder inside the Siamese Net to transfer the knowledge of the item embeddings to the item encoder. In some existing related works [23], the soft node/edge labels of the trained GNN model from node/edge classification are used for distillation. The reason of using node embeddings instead of soft edge labels as targets in the TA/CA task is as follows. In the Single-Task TA/CA graph, the overlap between the items in the catalog samples and the items in the audited pairs is limited. For example, in the sampled data of our experiments, although we have over 2MM audited TA/CA item pairs in total, and over 800K items in the Single-task TA/CA Graph, the overlapped item pairs between the Single-task TA/CA Graph and the audited pairs is less than 70K. Therefore, using node embeddings (800K) could help to fully leverage the graph knowledge.

We use Mean Square Error (MSE) loss in the TA/CA distillation to minimize the L2-distance between item embeddings and the item encoder's outputs. After distillation, we finetune the entire Siamese Net on the audited pairs. For a pair of input items, the output embeddings of item encoders are merged and concatenated with hand-crafted features. The concatenated embeddings are fed into a MLP layer for classification predictions. The entire pipeline is illustrated in Figure 4.

In the BV case, similarly we adopt a pairwise model which shares the same architecture as the BV baseline model for distillation. Inspired by [23], we use the distillation objective in Eq. (3) to jointly distill the knowledge from soft edge labels of a well-trained RGCN, and finetune with the BV true labels. In the constructed Single-task BV Graph in our experiments, we have more labeled edges than nodes (440K labeled edges vs. 22K nodes), so using soft edge labels could help to fully leverage the graph knowledge in the BV Graph. Using soft edge labels also aligns with the classification task of BV item pairs, so that the pairwise model can be trained end-to-end.

$$\mathcal{L} = \lambda \sum_{e \in E} \mathcal{L}_{finetune}(\hat{\mathbf{y}}_e, \mathbf{y}_e) + (1 - \lambda) \sum_{e \in E} \mathcal{L}_{distill}(\hat{\mathbf{y}}_e, \mathbf{z}_e) \quad (3)$$

λ is a weighting hyperparameter to weight the finetune loss and the distillation loss. $\mathcal{L}_{finetune}$ is a cross-entropy loss which takes the edge prediction $\hat{\mathbf{y}}_e$ and true edge label $\mathbf{y}_e$ as inputs. $\mathcal{L}_{distill}$ is

a KL-divergence loss which takes the edge prediction $\hat{\mathbf{y}}_e$ and the soft edge label $\mathbf{z}_e$ as inputs.

4 EXPERIMENTS

4.1 Datasets and Experimental Settings

We evaluate the proposed framework on TA/CA and BV problems against the pairwise baseline models, which are the state-of-the-art pairwise models. We use a catalog sample and audited pairwise datasets to create a Single-task TA/CA Graph and a Multi-task Graph for TA/CA in the scope of books, and a Single-task BV Graph for BV in the scope of groceries[1]. The Single-task TA/CA Graph contains 2.7MM edges with average degree 2.6. The BV Graph contains 440K edges with average degree 20. In the Multi-task Graph, there are 4.0MM edges with average degree 3.6. In the actual RGCN implementation, each edge relationship has a pair of relationships for two-way message passing. For example, the edges between Items and TA parents have relationship of `<Item, has_TA_parent, TA parent>` and `<TA parent, has_TA_child, Item>`. The symmetric and non-symmetric relationships are not distinguished for simplicity, since this design is able to cover both types. In total, there are 12 types of edge relationships.

The metrics used in the experiments are area under precision recall curve (PR-AUC) and recall at 95% precision (Recall@95).

4.2 Effectiveness Results

The results on inductive setting are shown in Table 1, Figure 6 and Figure 5 (a), tested on the both TA/CA and BV tasks. In Table 1, D-RGCN denotes distilled RGCN model. (S) and (M) denote using Single-task Graph and Multi-task Graph, respectively. D-RGCN$_p$ denotes the distilled RGCN model trained with pre-training strategy. Siamese and XLMR denote the Siamese Net and XLM-RoBERTa [14] model used as the baseline models for TA/CA and BV task. The last column shows whether the hand-crafted pairwise features are used for message passing in RGCN training and in distillation process (Figure 4). We can make the following observations. Firstly,

Table 1: Inductive results on both TA/CA and BV tasks.

Dataset	Model	PR-AUC	Recall@95	Pairwise Features
TA/CA	Siamese	0.800	0.015	No
	D-RGCN (S)	**0.922**	**0.450**	
	Siamese	0.904	0.337	Yes
	D-RGCN (S)	0.928	0.673	
	D-RGCN (M)	0.941	0.676	
	D-RGCN$_p$ (S)	<u>0.942</u>	**0.699**	
	D-RGCN$_p$ (M)	**0.946**	<u>0.691</u>	
BV	XLMR	0.775	0.041	Yes
	D-RGCN	**0.806**	**0.181**	

the distilled RGCN shows significant improvement over baseline method on both TA/CA and BV datasets. When pairwise features are used, the Recall@95 shows up to 33.6% improvement on the TA/CA dataset, and 14.0% on BV dataset. The PR-AUC has up to 2.4% improvement on the TA/CA dataset, and 3.1% improvement on BV dataset. Secondly, the pairwise features have huge positive impact on the model performance. Thirdly, using pre-training strategy further improves the D-RGCN performance consistently on both Single- and Multi-task Graph, for instance, by 1.4% in PR-AUC on Single-task Graph. Fourthly, Multi-task Graph helps to slightly improve the D-RGCN's PR-AUC performance.

Some additional experimental results on BV are shown in Figure 5. We can see from Figure 5 (b) and (c) that even when the RGCN model is not well-trained, the distilled RGCN still outperforms both the XLMR baseline and the RGCN edge classification, especially in Figure 5 (c), where the RGCN edge classification performance is even worse than the XLMR baseline. These results further exhibit the effectiveness of the proposed distillation method.

4.3 Graph Pruning Study

We further conduct graph pruning study on the effectiveness of the different relationships we include in the created heterogeneous graphs, and the results on the TA/CA dataset are presented in Table 2. Generally, using the Multi-task Graph shows the best performance over the rest of the settings, as when BN Graph is removed, the PR-AUC drops from 0.941 to 0.928, and the recall@95 also slightly drops. Additional observations can be made as follows. Firstly, when only TA/CA graph is used, removing (Item, CA parent) edges significantly drops the recall@95 from 0.673 to 0.637, which indicates that this type of edge is quite useful in the TA/CA problem. Secondly, we study the importance of (Item, TA parent) edges by gradually and randomly removing a certain percentage. As we can see, as we remove more (Item, TA parent) edges, the drop of recall@95 becomes larger. (Item, TA parent) edges have the most direct connections to the TA task, so it is reasonable to see the large decrease of performance when they are removed. Lastly, removing (TA parent, CA parent) edges increases the PR-AUC from 0.928 to 0.934, and recall@95 from 0.673 to 0.678, which indicates that this edge type may have negative contribution to the TA/CA problem. After this type of edge is removed from Multi-task Graph, we also observe a slight improvement. Judging only from the drops of recall@95 from the Single-task Graph performance, we can roughly conclude that the importance of the studied edge relationships is (Item, TA parent) > (Item, CA parent) > (CA parent, TA parent).

Table 2: Pruning Study of Different Relationship Types.

Model	PR-AUC	Recall@95	Graph Pruning
Siamese	0.904	0.337	N/A
D-RGCN (S)	0.928	0.673	N/A
	0.928	0.665	25% (Item, TA parent)
	0.927	0.663	50% (Item, TA parent)
	0.924	0.385	75% (Item, TA parent)
	0.931	0.637	100% (Item, CA parent)
	0.934	**0.678**	100% (TA parent, CA parent)
D-RGCN (M)	0.941	0.676	N/A
	0.942	**0.683**	100% (TA parent, CA parent)

The study of Table 2 suggest that the effectiveness of different relationships included in a heterogeneous graph should be studied case-by-case. Leveraging additional relationships may not be

[1]The datasets are samples of the Amazon catalog, which are non-representative to Amazon's production data.

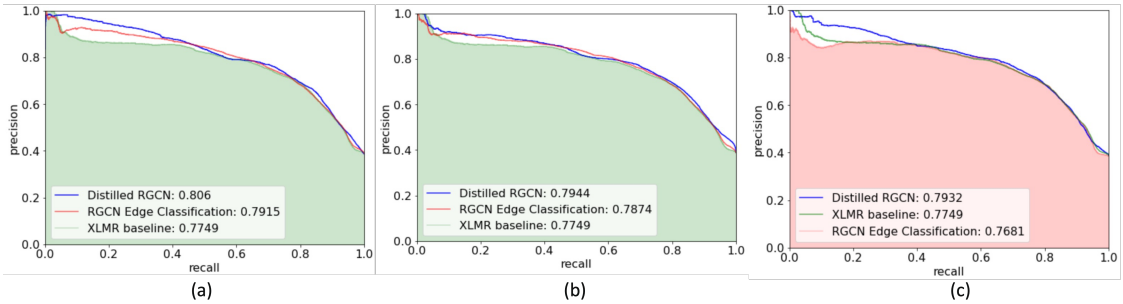

Figure 5: Additional results of BV. (a) corresponds to the results in Table 1. RGCN Edge Classification shows the testing performance of the RGCN model on labeled edges. Lower Edge Classification performance in (b) and (c) compared to (a) shows that the RGCN models trained in (b) and (c) are sub-optimal.

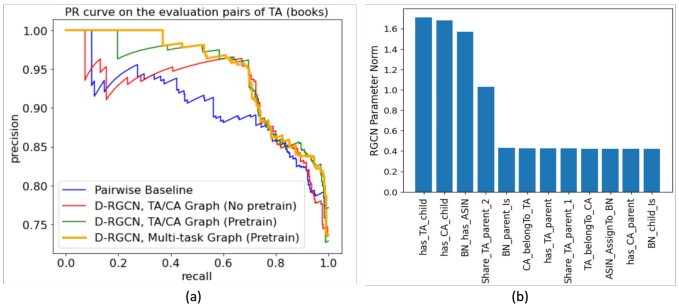

Figure 6: Experimental results of precision-recall curves on TA task (a), and relevance score of edge relationships (b).

guaranteed to help with the downstream problem. However, does there exist an indicator for the importance of each edge relationship towards a downstream problem? We plot the Frobenius norm of the relation weights $||\mathbf{W}_r^{(1)}||_F$ (from Eq. (1)) inside the well-trained RGCN model, and show them in Figure 6 (b). The x-axis shows the edge relationships and the y-axis shows the sorted Frobenius norm of the relation weights. Note that all relation weight matrices have the same dimensions. Note that the implementation of RGCN uses the reversed relationships in message passing, so each edge has a pair of relationships. The top-3 largest relationships are `has_TA_child`, `has_CA_child`, and `BN_has_Item`. Surprisingly, the top-2 aligns with our conclusion from Table 2. It indicates that using the Frobenius norm of the relation weights in RGCN might be a good indicator of the importance of different edge relationships.

5 RELATED WORK

GNNs Distillation. GNNs are a series of powerful neural models that show superior impact on a variety of graph mining research problems and applications [2, 3, 7–9, 11, 13, 16, 19, 22]. Distillation methods on GNNs are recent directions in GNNs research. Chen et al. [5] propose a self-distilling GNN model to adopt adaptive discrepancy retaining (ADR) regularizer to empower the transferability of knowledge that maintains high neighborhood discrepancy across GNNs layers. GLNN by Zhang et al. [23] shows that the performance of MLPs can be improved by large margins with GNNs knowledge distillation. GLNN has competitive accuracy, but infers faster than GNN model. G-CRD by Joshi et al. [12] uses contrastive learning to implicitly preserve global topology by aligning the student node embeddings to those of the teacher in a shared representation space. Inductive GNN is another research direction which draws much attention recently. Traditional semi-supervised graph classification tasks adopt transductive setting where all nodes and edges are observed during training. Recently, Qu et al. propose a Structured Proxy Network (SPN) [18] for inductive node classification. Furthermore, DEAL by Hao et al. [10] and Topology-Aware Correlation model (TACT) by Chen et al. [4] are two representative inductive link prediction methods.

Pre-training for GNNs. Recent advances in pre-training methods for Graph Neural Networks (GNNs) have shown significant improvements in graph representation learning tasks. One such method is the unsupervised pre-training method known as "Deep Graph Infomax" (DGI), introduced by Velickovic et al. [20] in 2018. DGI maximizes the mutual information between local node representations and the global graph representation by training a GNN encoder to predict the global graph representation using only local neighborhood information. Another notable pre-training method is the self-supervised "Graph Contrastive Coding" (GCC), proposed by Qiu et al. in 2020 [17]. GCC utilizes graph contrastive learning to maximize the similarity between augmented node embeddings while minimizing the similarity between non-augmented node embeddings. Both DGI and GCC have shown promising results in downstream tasks such as node classification and link prediction. More recently, the progress in pre-training methods on graphs using graph neural networks (GNNs) have shown promising results in various downstream tasks. One such method is PKCG introduced by Lv et al. [15]. PKCG explores whether pretraining techniques is able to benefit knowledge graph completion.

6 CONCLUSION

Graphs have shown potential in addressing complex catalog relationship problems. We investigate how to create multi-entity, multi-relationship graphs for handling multiple relationship problems in a self-supervised fashion. Furthermore, we introduce a GNN distillation approach that can transfer the knowledge learned by a GNN model trained on heterogeneous graphs to a pairwise model. The distilled model retains the same architecture as the pairwise baseline model, thereby facilitating a seamless replacement without major changes in the deployment logic and inference latency. Our experiments show that the multi-entity, multi-relationship graph is beneficial for individual catalog relationship problems, and the

appropriate pre-training technique can enhance supervised downstream tasks. The distillation method improves the baselines of multiple catalog relationship tasks significantly. This study opens up opportunities for future research to explore more generic questions of graph utilization for large e-commerce catalog problems.

REFERENCES

[1] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. 2016. Fully-convolutional siamese networks for object tracking. In *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part II 14*. Springer, 850–865.

[2] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. 2017. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine* 34, 4 (2017), 18–42.

[3] Dan Busbridge, Dane Sherburn, Pietro Cavallo, and Nils Y Hammerla. 2019. Relational graph attention networks. *arXiv preprint arXiv:1904.05811* (2019).

[4] Jiajun Chen, Huarui He, Feng Wu, and Jie Wang. 2021. Topology-aware correlations between relations for inductive link prediction in knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 6271–6278.

[5] Yuzhao Chen, Yatao Bian, Xi Xiao, Yu Rong, Tingyang Xu, and Junzhou Huang. 2020. On self-distilling graph neural network. *arXiv preprint arXiv:2011.02255* (2020).

[6] Boxin Du, Changhe Yuan, Robert Barton, Tal Neiman, and Hanghang Tong. 2021. Hypergraph pre-training with graph neural networks. *arXiv preprint arXiv:2105.10862* (2021).

[7] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of The Web Conference 2020*. 2331–2341.

[8] Anna Goldenberg, Alice X Zheng, Stephen E Fienberg, Edoardo M Airoldi, et al. 2010. A survey of statistical network models. *Foundations and Trends® in Machine Learning* 2, 2 (2010), 129–233.

[9] Palash Goyal, Sujit Rokka Chhetri, and Arquimedes Canedo. 2020. dyngraph2vec: Capturing network dynamics using dynamic graph representation learning. *Knowledge-Based Systems* 187 (2020), 104816.

[10] Yu Hao, Xin Cao, Yixiang Fang, Xike Xie, and Sibo Wang. 2020. Inductive link prediction for nodes having only attribute information. *arXiv preprint arXiv:2007.08053* (2020).

[11] Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. 2007. Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*. 56–65.

[12] Chaitanya K Joshi, Fayao Liu, Xu Xun, Jie Lin, and Chuan Sheng Foo. 2022. On representation knowledge distillation for graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[13] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[14] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[15] Xin Lv, Yankai Lin, Yixin Cao, Lei Hou, Juanzi Li, Zhiyuan Liu, Peng Li, and Jie Zhou. 2022. Do pre-trained models benefit knowledge graph completion? a reliable evaluation and a reasonable approach. Association for Computational Linguistics.

[16] Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*. 404–411.

[17] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 1150–1160.

[18] Meng Qu, Huiyu Cai, and Jian Tang. 2022. Neural structured prediction for inductive node classification. *arXiv preprint arXiv:2204.07524* (2022).

[19] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*. Springer, 593–607.

[20] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. *ICLR (Poster)* 2, 3 (2019), 4.

[21] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826* (2018).

[22] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. 2019. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 793–803.

[23] Shichang Zhang, Yozen Liu, Yizhou Sun, and Neil Shah. 2021. Graph-less neural networks: Teaching old mlps new tricks via distillation. *arXiv preprint arXiv:2110.08727* (2021).