MERLIN: Multimodal & Multilingual Embedding for Recommendations at Large-scale via Item Associations

Sambeet Tiady
Amazon.com
Bangalore, India
tiadys@amazon.com

Arihant Jain
Amazon.com
Bangalore, India
arihanta@amazon.com

Dweeen Rabius Sanny
Amazon.com
Bangalore, India
drsanny@amazon.com

Khushi Gupta
Amazon.com
Bangalore, India
ggkhushi@amazon.com

Srinivas Virinchi
Amazon.com
Bangalore, India
virins@amazon.com

Swapnil Gupta
Amazon.com
Bangalore, India
swapgupt@amazon.com

Anoop Saladi
Amazon.com
Bangalore, India
saladias@amazon.com

Deepak Gupta
Amazon.com
Bangalore, India
dgupt@amazon.com

Abstract

Product recommendations incentivize customers to make multi-unit purchases by surfacing relevant products, leading to lower cost per unit for e-commerce stores and lower prices for their customers. However, the humongous scale of products, implicit co-purchase asymmetry and variation in co-purchase behavior across different categories, are orthogonal problems to solve. To address these problems, we propose MERLIN (Multimodal & Multilingual Embedding for Recommendations at Large-scale via Item associations), a Graph Neural Network that generates product recommendations from a heterogeneous and directed product graph. We mine category associations to remove noisy product co-purchase associations, leading to higher quality recommendations. Leveraging product co-view relationships, we finetune SentenceBERT model for textual representation, and train a self-supervised knowledge distillation model to learn visual representation, which allows us to learn product representations which are multi-lingual and multi-modal in nature. We selectively align node embeddings leveraging co-viewed products. MERLIN model can handle node asymmetry by learning dual embeddings for each product, and can generate recommendations for cold-start products by employing catalog metadata such as title, category and image. Extensive offline experiments on internal and external datasets show that MERLIN model outperforms state-of-the-art baselines for node recommendation and link prediction task. We conduct ablations to quantify the impact of our model components and choices. Further, MERLIN model delivers significant improvement in sales measured through an A/B experiment.

CCS Concepts

• **Applied computing** → **Online shopping**; • **Computing methodologies** → **Neural networks**; • **Information systems** → *Retrieval models and ranking*; **Recommender systems**; **Online shopping**.

Keywords

Product Recommendation; Graph Neural Networks; Representation Learning; Link Prediction; Node Recommendation

CIKM '24, October 21–25, 2024, Boise, ID, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3680106

ACM Reference Format:
Sambeet Tiady, Arihant Jain, Dweeen Rabius Sanny, Khushi Gupta, Srinivas Virinchi, Swapnil Gupta, Anoop Saladi, and Deepak Gupta. 2024. MERLIN: Multimodal & Multilingual Embedding for Recommendations at Large-scale via Item Associations. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3627673.3680106

1 Introduction

Many e-commerce stores were launched with the primary objective of enabling customers to conveniently shop for a wide range of products at the best price. As they have grown to offer a vast catalog of products, the need to get fit and lean has become increasingly important in order to compete in the ultra-competitive e-commerce industry. Multi-unit purchases directly contribute to an e-commerce stores's bottom line, leading to lower costs per unit. Units per purchase (UPP), which measures percentage of purchases that have more than one unit, serves as a critical metric in this regard. Higher UPP leads to lower costs, which translates to lower prices for the customers. Product recommendations which surface relevant and related products to the base product being considered, reduce the cognitive load for customers, thereby nudging them to make multi-unit purchases and directly contribute to higher UPP.

Customers generally co-purchase three types of related items together: a) Complementary products that enhance the productivity of the other product, b) Adjacent products from related categories serving similar end needs, and c) Substitutes based on quantity, color or complex attributes like visual similarity. Co-purchase behavior varies across product categories, driven by a difference in end customer needs and the additional value they derive from co-purchasing these items together. For example, within electronics, the dominant customer need is to enhance a base product's productivity (screen guard for a smartphone), while for consumables, this behavior changes to adjacent products (bread and milk) or quantity substitutes. In apparel categories, it is more nuanced based on visual similarity (beach shirts) or visual compatibility (tan shoes and belts). Co-purchase relationships are also implicitly asymmetric, i.e., for a phone, we would like to recommend a phone case, but for a phone case, it may not be apt to recommend a phone.

Furthermore, e-commerce co-purchase data can be extremely noisy - customers purchase unrelated products such as electronic devices and cosmetics together, possibly due to high discounts and multiple shopping profiles. E-commerce stores solve for this customer need by building product recommendations and basket building experiences, built on top of relational customer data via pattern matching, or manual curation through hand-crafted rules and category intelligence. While successful in isolation, pattern matching or manual curation approaches are sub-optimal due to lower relevance of recommendations or lower coverage.

To address these problems, we propose - MERLIN (Multimodal and Multilingual Embedding for Recommendations at Large-scale via Item Associations), a Graph Neural Network model which generates product recommendations on a heterogeneous directed graph. We formulate this as a node recommendation problem given a set of query products. Our hetero-graph consists of products as nodes and edges corresponding to co-purchase or co-view relationship between the products. MERLIN framework can model asymmetry in co-purchase data by learning two representations for each product. We do not limit ourselves to complementary or substitute product recommendation, and depending on co-purchase behavior, we can recommend both. We learn multi-lingual and multi-modal product embeddings and selectively align them leveraging co-viewed products. Employing product catalog metadata allows us to generate recommendations for cold-start products. For noisy item relationships, we propose a pre-processing step requiring minimal human intervention leading to higher quality recommendations. We list out the contributions made by our paper below:

(1) We model co-purchase asymmetry using dual node representations and show that it improves the ability to predict link direction by 2%, compared to state-of-the-art baselines.
(2) We learn language-agnostic, multi-modal product embedding using a finetuned multi-lingual SentenceBERT and self-supervised ViT (Vision Transformers) model, and selectively align them using product co-view relationships.
(3) Extensive offline experiments show that MERLIN model outperforms state-of-the-art baselines by $10x$ in terms of Hits/MRR for node recommendation and by 13% in terms of AUC for link prediction. We conduct ablation studies to quantify the impact of our model components.
(4) Lastly, MERLIN model delivers significant improvement in sales (+1.5%) and UPP (+1.1%) over incumbent techniques, measured via an A/B experiment.

2 Related Work

Our problem conceptually relates to the node recommendation problem [10, 22, 26, 33, 32] in directed graphs. Initial work relied on random walk models to capture node relationships in directed graphs. However, these models cannot be applied to directed graphs as they cannot capture the direction of the edge. VERSE [29], HOPE [19], NERD [14] and APP [39] learn two embeddings for each node to preserve higher order proximity and node asymmetry in directed graphs. ATP [25] addresses the problem of question answering by embedding nodes of directed graphs by preserving node asymmetry. However, ATP is strictly restricted to directed acyclic graphs (DAGs), while real-world graphs are not acyclic. NERD [14] learns a pair of role-specific embeddings for each node using an alternating random walk strategy to capture edge strength and direction in directed graphs. All these models are transductive in nature and cannot be extended to inference task on unseen nodes.

With GNNs being superior compared to random walk models, the latest trend has seen a shift towards designing GNNs for directed graphs. DGCN [28], APPNP [16], and DiGraphIB [27] are GCN based models that capture first and second-order node proximity in directed graphs. Gravity GAE [24] (Graph Auto Encoder [15] based model) and DGGAN [40] (Generative Adversarial Network [8] based model) learn a pair of embeddings for each node. SEAL [37] proposes a new method to learn heuristics by extracting local subgraphs around each target link and using a GNN for link prediction. MagNet [38] proposes a GNN for directed graphs based on a complex Hermitian matrix to jointly model graph structure and node asymmetry. DIVINE [35] introduces the concept of virtual negative edges (VNEs), decides the number and locations of VNEs, then learns embeddings by exploiting both the signs and directions of edges. Although pre-training and contrastive learning is effective for CNNs and NLP, it is still under exploration for GNNs. Recent works like GraphCL [36] provide robust representation learning through various graph augmentations.

The field of self-supervised representation learning has garnered significant attention in recent years. Current research efforts primarily concentrate on two approaches: Contrastive learning and Knowledge distillation-based learning. Contrastive learning methods, such as [11, 6, 3] require the presence of both positive and negative samples to maximize the distance between their representations. Negative sampling helps improve the model's ability to discriminate between positive and negative samples [17, 20]. However, the quality [5] and quantity [18] of negative samples can significantly impact the downstream performance. Moreover, generating high-quality negative samples can be challenging. On the other hand, self-distillation or knowledge distillation-based methods, rely solely on positive pairs and distill the logits (outputs) of a teacher model into a student model. Notable examples of self-distillation approaches include BYOL [9], SimSiam [4], and DINO [2], which have eliminated the need of negative samples.

3 MERLIN Framework

Table 1: Notation

Notation	Description
P	set of products in catalog
C	set of product categories in catalog
$i \in P$	product i in catalog
X_i	input feature of product i
G	directed product graph
θ_i^s	source embedding of product i
θ_i^t	target embedding of product i
$q \in P$	query product
R_k^q	top-k related products for query product q
CP	product co-purchase pairs
CV	product co-view pairs
E_{cp}	product co-purchase edges
E_{cv}	product co-view edges

Let G be a directed product graph with products P as the nodes and directed edges corresponding to a co-purchase or co-view relationship between the products (refer section 3.1). Further, every product i ($i \in P, \forall i$) has an input feature X_i from the product catalog metadata. Given a query node[1] q, the goal is to recommend R_k^q, top-k related products that have a high likelihood to be bought together with q. Table 1 introduces the notation that we will use in the rest of the paper.

3.1 Product graph construction

We show the proposed framework in Figure 1. The goal is to recommend the top-k related products that are likely to be bought with a given query product. Given a set of product co-purchase pairs CP, we create the co-purchase edges $E_{cp} = \{(u, v) | \forall\, u, v \in P \wedge uR_{cp}v\}$. We use E_{cp} to model the product co-purchase likelihood. However, E_{cp} can be prone to selection bias due to inventory issues and leads to lower coverage than desired. Therefore, we use CV to create the set of co-view edges $E_{cv} = \{(u, v) | \forall\, u, v \in P \wedge uR_{cv}v\}$. We fix the edge direction based on the median price of product categories, i.e. edge is directed from a higher priced category to a lower priced one. This can be easily replaced with other signals like product views, popularity, purchase time, etc. This allows us to construct a directed product graph $G = (P, \{E_{cp} \cup E_{cv}\})$, which contains both co-purchase and co-view relationships between products. To remove noisy co-purchase relationships, the framework computes category association scores using Normalized Pointwise Mutual Information ($NPMI$) and matrix factorization, which we describe next. The node features are initialized using fine-tuned embeddings from a Sentence-BERT [23] model for product category and title, concatenated with image embeddings from a self-supervised ViT [7] model, described in subsequent sections.

3.2 Graph Pre-processing using Category Association Scores

Item purchase relationships can be noisy in e-commerce store data, possibly due to a) customers purchasing unrelated products such as electronics and cosmetics at discounts during sale or multiple shopping personas attributed to the same customer and b) complementary categories such as screen protectors and smartphones, have lower observed co-purchase occurrence due to less popularity. To mitigate this, we compute a category association score based on Normalised Pointwise Mutual Information ($NPMI$ [1]) as follows:

$$NPMI_{(a,b)} = -\log\{P(a,b)/P(a) * P(b)\} / \log\{P(a,b)\} \tag{1}$$

where, $P(a, b)$ = probability of co-purchase from category a and b, and $P(x)$ = probability of purchase from category x. To avoid spurious category associations, we only calculate $NPMI$ score for category combinations, where co-purchase count is above a minimum threshold, which further leads to sparsity and unobserved associations (~3100 associations). To mitigate this, we learn an embedding for each category $h_a | \forall\, a \in C$, such that cosine similarity between embeddings of any two categories would reconstruct their $NPMI$ score. We minimise the following objective to learn the category embeddings: $loss_{cat} = \sum_{(a,b)\in C} (\cos(h_a, h_b) - NPMI_{(a,b)})^2$, where a and b are categories in the set of product categories C, for which $NPMI_{(a,b)}$ exists and h_a and h_b are the embeddings learnt for categories a and b respectively. We use the reconstructed category association scores as $AS_{(a,b)} = \cos(h_a, h_b)$, to filter out noisy co-purchase relationships where $AS_{(a,b)}$ is less than 0.2 and end up with ~4500 category associations.

[1]We use product and node interchangeably in this paper based on the context.

3.3 Multi-lingual Text Embeddings using Finetuned SentenceBERT

Fine-tuning language models for specific tasks incorporates domain-specific knowledge from the input text, which can improve the downstream performance significantly. With the same motivation, we leverage a SentenceBERT model fine-tuned on a related task for our data set, for representing textual product attributes. To extend MERLIN model to handle multi-lingual text across multiple marketplaces, we used a multilingual version of the Sentence-BERT model. To prepare the training examples, we use the product relationships in the graph. For anchor-positive pairs (a, p), we sample product co-view pairs CV from G. For negatives (n), we explore three sampling schemes: a) sample from products co-purchased with the anchor (FTCopurchase), b) mix of random negatives from G and products from the same category as the anchor (FTCategory) and c) additionally mine hard negatives from co-viewed products, when $CoviewCount(a, n) < CoviewCount(a, p)$ and $CoviewCount(a, n) >= 10$ (FTCoview). We then use the product category and title of a, p and n to create triplets and fine-tune Sentence-BERT [23] with the following objective:

$$loss_{ft} = -\sum_{(a,p)\in E_{cv}} \left(\sum_{\substack{n_s=1 \\ n\sim P_r(cat_a)}}^{n_k} max(0, M_{ft} - \cos(a,p) + \cos(a,n)) \right)$$

where, a and p represent anchor and positive in a triplet sourced from co-view pairs E_{cv}, M_{ft} represents margin used in cosine margin loss (M_{ft} is set to 0.5), and n represents negative in a triplet, randomly sampled n_k times.

Further, to improve cross-lingual text representation, we propose an augmentation strategy which leverages products which are sold in multiple countries, and replaces 20% of English titles with its title in another language (FTCoviewML). This aligns text representations in different languages and facilitates robust model training. We fine-tune our model and report the results for 8 marketplaces consisting of English, Spanish and Portuguese languages. The results of these fine-tuning experiments are reported in Section 4.2.3.

3.4 Visual Similarity Learning using Self-Supervised Training

In addition to the textual embeddings from fine-tuned Sentence-BERT, we incorporate visual similarity to further improve the recommendation quality. The visual representation learning approach is inspired by DINO [2] with a ViT [7] backbone, which employs a self-supervised teacher-student framework [41] that does not require negatives. We leverage the inherent product co-view relationships as positives along with augmentation strategies defined in SKILL [31]. The primary intuition is that if two products are often co-viewed by customers and differ in certain attributes like color, pose or orientation, the model would become invariant to those features and learn to focus on the more discriminating features like style, pattern and texture.

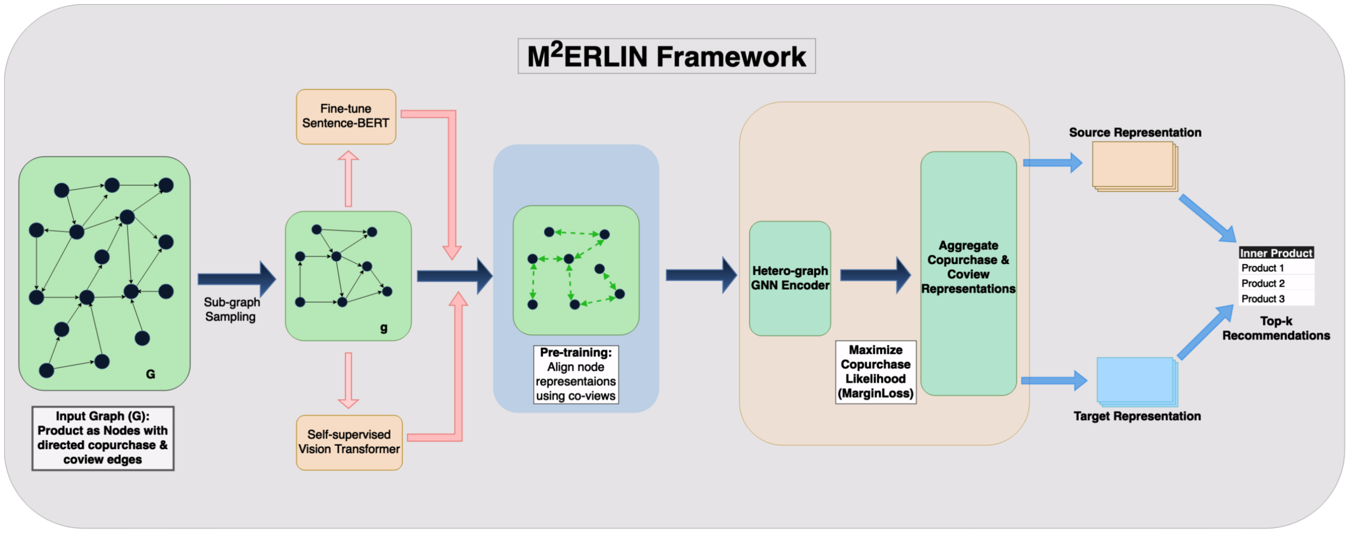
M²ERLIN Framework
Fine-tune Sentence-BERT
Sub-graph Sampling
G
g
Input Graph (G): Product as Nodes with directed copurchase & coview edges
Self-supervised Vision Transformer
Pre-training: Align node representaions using co-views
Hetero-graph GNN Encoder
Aggregate Copurchase & Coview Representations
Maximize Copurchase Likelihood (MarginLoss)
Source Representation
Target Representation
Inner Product
Product 1
Product 2
Product 3
Top-k Recommendations

Figure 1: Proposed Framework for MERLIN - For each node A in the product graph, our model produces a source (s) and target (t) representation as shown.

Specifically, for a given co-viewed product pair i and j, we pass their images (x'_i, x'_j) as the positive pair. We pass these through the global and local augmentation functions $\mathcal{G}$ and $\mathcal{A}$, respectively. The global augmentation contains the high resolution image with various transformations like *RandomCrop*, *RandomGray*, *ColorJitter* and *GaussianBlur*, which help the model learn features that are not affected by color changes or blurring, and the local augmentations contain a small low resolution random crop with the same global transformation applied to it. The augmented images $(\mathcal{G}(x'_i), \mathcal{A}(x''_i))$ are then fed to the student model $\mathcal{S}_\theta$ to obtain the output distributions o^s_i. Similarly, the teacher model $\mathcal{T}_\phi$ receives only the high-resolution augmented images $\mathcal{G}(x'_j)$ and produces the target distributions o^t_j. The teacher model parameters ϕ are updated as an exponential moving average of the student parameters θ, similar to BYOL [9]. We optimize the cross-entropy loss between the student's predictions o^s_i and the teacher's targets o^t_j:

$$\mathcal{L}(\theta, \phi) = \sum_{(x'_i, x'_j)} -o^t_j \log o^s_i \tag{2}$$

This self-supervised visual similarity learning, combined with the textual embeddings, allows the model to learn comprehensive multi-modal product representations for recommendation. We present a detailed analysis comparing our proposed method against a) text-only model (TextOnly), b) models pretrained on ImageNet (PreTrained), and c) ViT model trained on image variants of the same product (SameProduct) in Sec. 4.2.3.

3.5 Selective Alignment of Node Representations

To further improve the product representations, we propose a graph pre-training step based on the intuition that not all similar products are semantically or visually similar. This involves aligning the node representations utilizing a co-view sub-graph of G and predicting the likelihood of co-view relationships (CoviewAlignment). After CoviewAlignment, products pairs with co-view edges between them, or products with overlapping co-view neighbours, are closer to each other in the embedding space. This alignment allows us to selectively align node representations based on their relationships. The loss for the co-view prediction task is as follows:

$$loss_{pt} = -\left\{ \sum_{(u,v)\in E_{cv}} \left(\sum_{\substack{n_s=1 \\ z\sim P_r(P), u\neq z}}^{n_k} max(0, M_{pt} - \theta_u \cdot \theta_v + \theta_u \cdot \theta_z) \right) \right\}$$

where, u and v are co-viewed product pairs sampled from E_{cv}, M_{pt} is the margin used in the loss (set to 1.0), and z is a negative product, randomly sampled n_k times. We present a detailed analysis on pre-training in Sec. 4.2.3.

3.6 Proposed GNN model

We first describe the product embedding generation procedure, i.e. forward pass of the model assuming that the model is already trained (Section 3.6.1). We then describe how we generate product recommendations which takes care of assymmetry in co-purchase data (Section 3.6.2). Finally, we describe how the model is trained using back-propagation in Section 3.6.3.

3.6.1 ***Product Embedding Generation:*** We consider a $l = \{1...L\}$ layer MERLIN model, having weight and bias parameters $\{W^l, B^l\}$ for each layer l. Let $(h^s_u)^l$ and $(h^t_u)^l$ denote the source and target representation of node u in the l^{th} layer respectively. For each node u, 1) the source embedding of u, h^s_u, is similar to the target embedding of its co-purchase and co-view out-neighbors, 2) the target embedding of u, h^t_u, is similar to the source embedding of co-purchase and co-view in-neighbors. Consequently, by employing both h^s_u and h^t_u, we are able to jointly model edge strength and direction as shown in Equation 3.

$$(h_u^s)^l = FC_l \left[\sum_{(u,v)\in E_{cp}} (h_v^t)^{l-1} W_{cp}^l, \sum_{(u,v)\in E_{cv}} (h_v^t)^{l-1} W_{cv}^l \right] \quad (3)$$

$$(h_v^t)^l = FC_l \left[\sum_{(u,v)\in E_{cp}} (h_u^s)^{l-1} W_{cp}^l, \sum_{(u,v)\in E_{cv}} (h_u^s)^{l-1} W_{cv}^l \right] \quad (4)$$

where, FC_l corresponds to a fully connected layer at layer l. For a layer l, we aggregate $(h_v^t)^{l-1}$ from its co-purchase and co-view out neighbors and pass them through a fully connected layer FC_l to generate $(h_u^s)^l$. We repeat a similar process for the target embedding. We repeat this process for L layers to generate the final source and target embedding of all nodes $\{\theta_u^s, \theta_u^t\}\ \forall u \in P$.

3.6.2 ***Product recommendation:*** Given a query product q belonging to a product category C_q, we use θ_q^s, the source embedding of q, to perform the nearest neighbor lookup in the target embedding space of all the products to recommend a top-k set of related products denoted by R_k^q. Specifically, for a query product $q \in P$, we compute a relevance score with respect to a candidate product $v \in P$ as follows: $rel(q,v) = (\theta_q^s)^\intercal (\theta_v^t)$ Observe that $rel(q,v) \neq rel(v,q)$, which helps capture product asymmetry.

3.6.3 ***Learning MERLIN parameters:*** In order to train the model in an unsupervised manner, we employ a margin loss which needs to be minimized. Specifically, the loss function is as follows:

$$loss_{tr} = -\left\{ \sum_{(u,v)\in E_{cp}} \left(\sum_{\substack{n_s=1 \\ z\sim\ P_r(P), u\neq z}}^{n_k} max(0, M_{tr} - \theta_u^s \cdot \theta_v^t + \theta_u^s \cdot \theta_z^t) \right) \right\}$$

Specifically, M_{tr} refers to the margin set to 1.0, u is a base product, v is a co-purchase neighbor and z is a randomly sampled product which acts as a negative to u and v pair. θ_u^s, θ_v^t, and θ_z^t represent the source embedding of u, target embedding of v, and target embedding of z, respectively. The loss function maximizes the difference between the score of co-purchase neighbors u and v, and a random negative neighbor z w.r.t the base node u, and ensures the separation is at least M_{tr}. We repeat this for n_k negative samples for each base node. We discuss the results in the next section.

4 Experiments

In this section, we demonstrate the effectiveness of MERLIN model using a comprehensive set of experiments. Specifically, we aim to answer the following questions: **E1**) How effective is MERLIN model on the primary task of node recommendation? **E2**) How effective is the MERLIN model on the link prediction task, and can it capture co-purchase asymmetry? **E3**) Are product embeddings learned through MERLIN model effective across different marketplaces, languages and modalities? **E4**) How do the recommendations generated by MERLIN perform in an online experiment? We introduce the experimental setup, dataset and baselines and follow it up with experiments to answer the above questions.

4.1 Experimental Setup

4.1.1 ***Dataset:*** The dataset used for **E1, E2** consists of a large product graph from a major marketplace (containing 6.5MM nodes, 9MM co-purchase edges, and 65MM co-view edges across all product categories), to facilitate comparison with internal baselines which are not present in all marketplaces. For **E3**, results were averaged across 8 marketplaces spanning English, Spanish and Portuguese languages. To facilitate easier comparison across the multiple data sets and to maintain confidentiality, all results are presented as relative lifts compared to the worst performing baseline as 1.00x. Additionally, for E1 and E2, we report the results on an external data set - a randomly sampled sub-graph of *ogbl-citation2*, consisting of 154K nodes and 210K directed edges. We use the average of word2vec embedding of the paper titles to represent the nodes. While it is a citation data set, it shares similar properties to ours i.e. directed edges, and also helps prove the effectiveness of our proposed framework beyond the product recommendation use-case.

4.1.2 ***Baselines:*** For the node recommendation task, the competitive baselines used were NERD [14] and DIVINE [35]. We also use two internal baselines: IB_1 and IB_2, which are curated using handcrafted rules on top of catalog and product relationships via pattern matching. For the link prediction task, the competitive baselines were DIVINE [35], SEAL [37], and MagNet [38].

4.1.3 ***Implementation Details:*** We implemented MERLIN model using DGL [34] and PyTorch [21] on p3, g5 and r6 EC2 instances. For fine-tuning Sentence-BERT model (*distiluse-base-multilingual-cased-v2*), we used a learning rate of 1e-5 with a margin of 0.5 for 10 epochs. For the CoviewAlignment and co-purchase prediction task, we set the learning rate to 1e-3, with a margin equal to 1, and train for a maximum of 20 epochs with early stopping employed on the validation dataset. For scalability during training and inference, we employ k-hop mini-batch sub-graph sampling by first sampling 10000 source nodes and then sampling their neighbours recursively up to 2-hops, with a maximum of 10 co-view neighbours and 20 co-purchase neighbours at each hop. For CoviewAlignment step, we used a 1-layer GCN model to align the 768 dimensional text embeddings and 256 dimensional image embeddings, whereas, for co-purchase prediction, we used a 2-layer GCN model with the final product embedding size of 128. We use FAISS [13] to perform an efficient nearest-neighbor lookup. The flexibility of our framework enables us to use any state-of-the-art GNN layer like GraphSage [12], GAT [30] etc., with slight modifications. We implemented baselines (DIVINE [35], SEAL [37], NERD [14] and MagNet [38]) using the original codes provided by the authors. We set the final node embedding dimensions to 256 for DIVINE [35] and 128 for other baselines. For DIVINE [35], we use the $STNE$ mode as the embedding generation algorithm.

4.1.4 ***Evaluation:*** We train all the models on randomly sampled training edges (70%) and tune the hyper-parameters on validation edges (10%). Finally, we report metrics on a held-out test set (20%). For the node recommendation task, we evaluated model performance using a) Hit Ratio (Hits@1,3,5) and b) Mean Reciprocal Rank (MRR@10). Due to the computational expense of evaluating using same metrics for link prediction on the entire graph, we compared our model to baselines using randomly sampled negatives: a) Hits@3 with 10 negative samples, b) MRR@10 with 10 negative samples, c) ability to rank positive links higher than 10 negative samples (AUC-random), and d) ability to predict link direction (AUC-dir), i.e. ranking positive links higher than negative links, obtained by inverting the direction of positive links. As mentioned before, all the results are relative and absolute numbers are not presented for better interpretation and confidentiality reasons.

4.2 [E1, E2, E3] Results

4.2.1 ***[E1] Node Recommendation.*** Table 2 compares the MERLIN model against the node recommendation baselines - NERD [14], DIVINE [35], and two internal baselines, IB_1 - manually curated using hand-crafted rules and category intelligence on top of catalog, and IB_2 - built on top of product relationships data via pattern matching. MERLIN model improves on MRR@10 by over +100% compared to internal baselines IB_1 and IB_2. It also outperforms NERD [2] and DIVINE [35] by up to 10x on Hits@k and MRR@10 metrics. Similarly, on the external data set (Table 3), MERLIN outperforms NERD and DIVINE by up to 10x, answering E1.

Table 2: Hits@k (k=1,3,5) and MRR@10 for MERLIN model compared to internal and external baselines for the internal data set.

Model	Hits@1	Hits@3	Hits@5	MRR@10
NERD	1.00x	1.00x	1.00x	1.00x
DIVINE	1.05x	1.11x	1.19x	1.10x
IB_1	4.67x	4.46x	4.46x	4.44x
IB_2	6.08x	5.57x	5.01x	4.95x
MERLIN	**11.05x**	**10.63x**	**10.38x**	**10.05x**

Table 3: Hits@k (k=1,3,5) and MRR@10 for MERLIN model compared to NERD and DIVINE for ogbl-citation2.

Model	Hits@1	Hits@3	Hits@5	MRR@10
NERD	1.00x	1.00x	1.00x	1.00x
DIVINE	1.00x	2.81x	4.48x	2.24x
MERLIN	**10.55x**	**8.69x**	**7.17x**	**7.92x**

4.2.2 ***[E2] Link prediction.*** Since SEAL [37], MagNet [38], and DIVINE [35] are algorithms for homogeneous graphs, for fair comparison, all models were trained on a homogeneous sub-graph with only co-purchase edges. The results in Table 4 show our model outperforming other baselines across all evaluation metrics. Compared to DIVINE, our model improves AUC-random by up to +13% and MRR@10 by up to 23%. For AUC-dir, we see a +1.5% lift over MagNet. Note that, as SEAL does not consider asymmetry, AUC-dir is exactly 0.5. These benefits are also observed on the external data set in Table 5, with 12% improvements in AUC-random and AUC-dir over DIVINE. The dual embeddings capturing edge direction are key to the lift in AUC-dir. This answers E2.

[2]https://git.l3s.uni-hannover.de/khosla/nerd/-/tree/master/

Table 4: Evaluation metrics of MERLIN model compared to SEAL, MagNet and DIVINE for internal data set.

Model	Hits@3	MRR@10	AUC-random	AUC-dir
DIVINE	1.00x	1.00x	1.00x	1.00x
SEAL	1.09x	1.17x	1.10x	-
MagNet	1.13x	1.13x	1.11x	1.01x
MERLIN	**1.14x**	**1.23x**	**1.13x**	**1.02x**

Table 5: Evaluation metrics of MERLIN model compared to SEAL, MagNet and DIVINE for ogbl-citation2.

Model	Hits@3	MRR@10	AUC-random	AUC-dir
DIVINE	1.00x	1.00x	1.00x	1.00x
SEAL	1.07x	1.10x	1.07x	-
MagNet	1.18x	1.19x	1.10x	1.05x
MERLIN	**1.24x**	**1.38x**	**1.12x**	**1.12x**

4.2.3 ***[E3] Effectiveness of Embeddings Learnt using MERLIN.*** This section systematically examines the importance of MERLIN's key components: (a) fine-tuning the multi-lingual Sentence-BERT model, b) self-supervised learning of visual representations, and (c) aligning node representations using co-views.

Fine-tuning Multilingual SentenceBERT Model: This experiment quantifies the impact of using a fine-tuned Sentence-BERT model for textual product representations, compared to a pre-trained SBERT model without fine-tuning (*PreTrained*). Within the fine-tuning process, three different negative sampling schemes while creating triplets: a) *FTCopurchase*, b) *FTCategory* and c) *FTCoview* were compared with *FTCoviewML* (multi-lingual augmentation strategy applied to FTCoview). The results are reported in Table 6. The best performance was observed with the *FTCoviewML* model, which showed a +16% lift in MRR@10 compared to the *PreTrained* model. Creating triplets by sampling hard negatives from co-view associations proves to be an effective strategy, and so does aligning language representations, as it allows the model to better differentiate between subtle product aspects and learn robust representations.

Table 6: Results comparing Hits@k and MRR@10 for PreTrained, FTCategory, FTCoview, FTCoviewML and FTCopurchase Sentence-BERT models

Sentence-BERT	Hits@1	Hits@3	Hits@5	MRR@10
PreTrained	1.00x	1.00x	1.00x	1.00x
FTCopurchase	1.07x	1.04x	1.02x	1.03x
FTCategory	1.11x	1.11x	1.10x	1.11x
FTCoview	1.15x	1.14x	1.13x	1.13x
FTCoviewML	**1.18x**	**1.15x**	**1.14x**	**1.16x**

Self-supervised Visual Similarity Learning: We evaluated the performance of our proposed visual representation approach

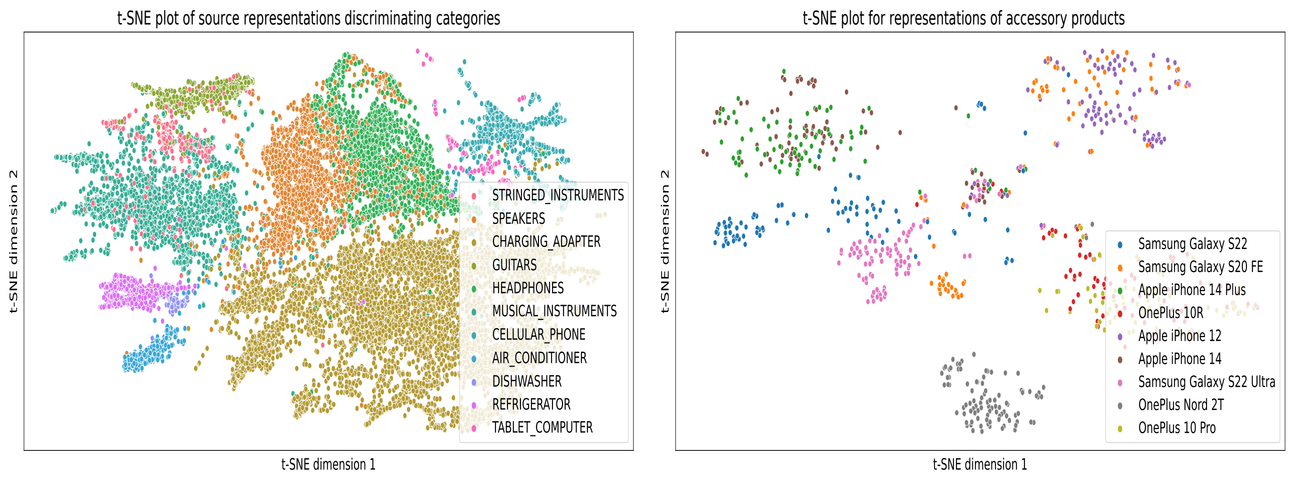

Figure 2: Figure showing 2D t-SNE plots of product representation learnt using MERLIN , showing the ability to differentiate between a) various product categories (left) and b) complementary products of different base products (right).

against several baselines: a) Text-only models (*TextOnly*), b) ImageNet pre-trained ViT model (*PreTrained*), c) ViT model trained on image variants of the same product (*SameProduct*) and d) ViT model trained on co-viewed product pairs (*CoviewPair*). As shown in Table 7, the *SameProduct* and *CoviewPair* models achieved similar lifts, with a +6% improvement in MRR@10 compared to *PreTrained*, and a +10% improvement over *TextOnly*. Further analysis revealed that the *CoviewPair* model performed better (+4%) than *SameProduct* model on predicting substitutes (same category co-purchases), but marginally underperformed (-1%) *SameProduct* model on complementary products.

Table 7: Results comparing Hits@k and MRR@10 for TextOnly, PreTrained, SameProduct and CoviewPair ViT models

ViT model	Hits@1	Hits@3	Hits@5	MRR@10
TextOnly	1.00x	1.00x	1.00x	1.00x
PreTrained	1.04x	1.04x	1.04x	1.04x
SameProduct	**1.08x**	**1.09x**	**1.11x**	**1.10x**
CoviewPair	**1.08x**	**1.09x**	1.10x	**1.10x**

Selective Alignment of Node Representations (CoviewAlignment): We evaluated pre-training the MERLIN model with a node representation alignment task, named *CoviewAlignment*, as discussed in Section 3.5. Table 8 shows a 10% lift in MRR@10 when using the *CoviewAlignment* pre-training task, which selectively aligns node representations based on product co-views, compared to the MERLIN model without any pre-training. This answers E3.

4.3 [E4] Online Experiment

The online performance of the MERLIN model was evaluated through a 4-week A/B test across 25 product categories and 36MM customer sessions in a large marketplace within our e-commerce stores. We surface the static item recommendations generated using MERLIN model on the product page as treatment, and compare to a control group which uses the internal baseline IB_2. We observed a statistically significant lift (p-value < 0.05) of +1.5% in product sales and +1.1% improvement in UPP, with minimal latency impact (<6ms). These results exhibit that product recommendations generated from MERLIN model can significantly improve customer shopping experience. This answers E4. Finally, we evaluate the product embeddings qualitatively using t-SNE in Figure 2, where we find that our product embeddings have the ability to successfully discriminate between a) various product categories (left) and b) complementary products of different base products (right).

Table 8: Effect of CoviewAlignment on Hits@k and MRR@10 compared to no pre-training

Pre-training	Hits@1	Hits@3	Hits@5	MRR@10
No Pre-training	1.00x	1.00x	1.00x	1.00x
CoviewAlignment	**1.13x**	**1.09x**	**1.09x**	**1.10x**

5 Conclusion and Future Work

In this paper, we present MERLIN, a GNN-based model which generates product recommendations on a heterogeneous directed graph. It models co-purchase asymmetry using dual node representations, which enhances the ability to predict link direction. We also propose a graph pre-processing step based on NPMI and matrix completion to improve recommendation quality. We also learn language-agnostic, multi-modal product embedding using finetuned SentenceBERT and self-supervised ViT models, and selectively align them using product co-view relationships. Comprehensive experiments and ablations, demonstrate the efficacy and efficiency of our model and its components. Future work will explore incorporating product side information using product attributes, compatible product pair identification and re-ranking recommendations based on customer session context.

References

[1] Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction.

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning* (ICML'20) Article 149. JMLR.org, 11 pages.

[4] Xinlei Chen and Kaiming He. 2020. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*.

[5] Guanyi Chu, Xiao Wang, Chuan Shi, and Xunqiang Jiang. 2021. Cuco: graph representation with curriculum contrastive learning. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. Zhi-Hua Zhou, (Ed.) Main Track. International Joint Conferences on Artificial Intelligence Organization, (Aug. 2021), 2300–2306. DOI: 10.24963/ijcai.2021/317.

[6] Alexey Dosovitskiy et al. 2020. An image is worth 16x16 words: transformers for image recognition at scale. In *International Conference on Learning Representations*.

[7] Alexey Dosovitskiy et al. 2020. An image is worth 16x16 words: transformers for image recognition at scale. In.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.

[9] Jean-Bastien Grill et al. 2020. Bootstrap your own latent: A new approach to self-supervised learning. *CoRR*, abs/2006.07733. https://arxiv.org/abs/2006.07733 arXiv: 2006.07733.

[10] Aditya Grover and Jure Leskovec. 2016. Node2vec: scalable feature learning for networks. In *22nd ACM SIGKDD*, 855–864.

[11] Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 2. IEEE, 1735–1742.

[12] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *NIPS*, 30.

[13] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7, 3, 535–547.

[14] Megha Khosla, Jurek Leonhardt, Wolfgang Nejdl, and Avishek Anand. 2019. Node representation learning for directed graphs. In *ECML PKDD*, 395–411.

[15] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*.

[16] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Predict then propagate: graph neural networks meet personalized pagerank. *ICLR*.

[17] Dugang Liu, Pengxiang Cheng, Zhenhua Dong, Xiuqiang He, Weike Pan, and Zhong Ming. 2020. A general knowledge distillation framework for counterfactual recommendation via uniform data. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR '20). Association for Computing Machinery, Virtual Event, China, 831–840. ISBN: 9781450380164. DOI: 10.1145/3397271.3401083.

[18] Kento Nozawa and Issei Sato. 2021. Understanding negative samples in instance discriminative self-supervised representation learning. *CoRR*, abs/2102.06866. https://arxiv.org/abs/2102.06866 arXiv: 2102.06866.

[19] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *SIGKDD*, 1105–1114.

[20] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. 2019. Relational knowledge distillation. *CoRR*, abs/1904.05068. http://arxiv.org/abs/1904.05068 arXiv: 1904.05068.

[21] Adam Paszke et al. 2019. Pytorch: an imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[22] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: online learning of social representations. In *SIGKDD*, 701–710.

[23] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: sentence embeddings using siamese bert-networks. (2019). arXiv: 1908.10084 [cs.CL].

[24] Guillaume Salha, Stratis Limnios, Romain Hennequin, Viet-Anh Tran, and Michalis Vazirgiannis. 2019. Gravity-inspired graph autoencoders for directed link prediction. In *CIKM*, 589–598.

[25] Jiankai Sun, Bortik Bandyopadhyay, Armin Bashizade, Jiongqian Liang, P Sadayappan, and Srinivasan Parthasarathy. 2019. Atp: directed graph embedding with asymmetric transitivity preservation. In *AAAI* number 01. Vol. 33, 265–272.

[26] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: large-scale information network embedding. In *WWW*, 1067–1077.

[27] Zekun Tong, Yuxuan Liang, Changsheng Sun, Xinke Li, David Rosenblum, and Andrew Lim. 2020. Digraph inception convolutional networks. *NIPS*, 33, 17907–17918.

[28] Zekun Tong, Yuxuan Liang, Changsheng Sun, David S Rosenblum, and Andrew Lim. 2020. Directed graph convolutional network. *arXiv preprint arXiv:2004.13970*.

[29] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, and Emmanuel Müller. 2018. Verse: versatile graph embeddings from similarity measures. In *Proceedings of the 2018 World Wide Web Conference*, 539–548.

[30] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks.

[31] Vinay Kumar Verma, Dween Rabius Sanny, Prateek Sircar, Shreyas Sunil Kulkarni, Deepak Gupta, and Abhishek Singh. 2023. Skill: skipping color and label landscape: self supervised design representations for products in e-commerce. In *CVPR 2023*. https://www.amazon.science/publications/skill-skipping-color-and-label-landscape-self-supervised-design-representations-for-products-in-e-commerce.

[32] Srinivas Virinchi and Anoop S V K K Saladi. 2023. Blade: biased neighborhood sampling based graph neural network for directed graphs. In *WSDM 2023*. https://www.amazon.science/publications/blade-biased-neighborhood-sampling-based-graph-neural-network-for-directed-graphs.

[33] Srinivas Virinchi, Anoop S V K K Saladi, and Abhirup Mondal. 2022. Recommending related products using graph neural networks in directed graphs. In *ECML-PKDD 2022*. https://www.amazon.science/publications/recommending-related-products-using-graph-neural-networks-in-directed-graphs.

[34] Minjie Wang et al. 2019. Deep graph library: a graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*.

[35] Hyunsik Yoo, Yeon-Chang Lee, Kijung Shin, and Sang-Wook Kim. 2022. Directed network embedding with virtual negative edges. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining* (WSDM '22). Association for Computing Machinery, Virtual Event, AZ, USA, 1291–1299. ISBN: 9781450391320. DOI: 10.1145/3488560.3498470.

[36] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33, 5812–5823.

[37] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31.

[38] Xitong Zhang, Yixuan He, Nathan Brugnone, Michael Perlmutter, and Matthew Hirn. 2021. Magnet: a neural network for directed graphs. *NIPS*, 34.

[39] Chang Zhou, Yuqiong Liu, Xiaofei Liu, Zhongyi Liu, and Jun Gao. 2017. Scalable graph embedding for asymmetric proximity. In *AAAI* number 1. Vol. 31.

[40] Shijie Zhu, Jianxin Li, Hao Peng, Senzhang Wang, Philip S Yu, and Lifang He. 2021. Adversarial directed graph embedding. *AAAI*.

[41] Matthieu Zimmer, Paolo Viappiani, and Paul Weng. 2014. Teacher-student framework: a reinforcement learning approach. In (May 2014).