HybridGCN: An Integrative Model for Scalable Recommender Systems with Knowledge Graph and Graph Neural Networks

Dang-Anh-Khoa Nguyen[1], Sang Kha[2], Thanh-Van Le*[3]
Ho Chi Minh City University of Technology (HCMUT), 268 Ly Thuong Kiet, District 10, Ho Chi Minh City, Vietnam[1,2,3]
Vietnam National University Ho Chi Minh City (VNU-HCM), Ho Chi Minh City, Vietnam[1,2,3]

***Abstract*—Graph Neural Networks (GNNs) have emerged as a state-of-the-art approach in building modern Recommender Systems (RS). By leveraging the complex relationships among items, users, and their attributes, which can be represented as a Knowledge Graph (KG), these models can explore implicit semantic sub-structures within graphs, thereby enhancing the learning of user and item representations. In this paper, we propose an end-to-end architectural framework for developing recommendation models based on GNNs and KGs, namely HybridGCN. Our proposed methodologies aim to address three main challenges: (1) making graph-based RS scalable on large-scale datasets, (2) constructing domain-specific KGs from unstructured data sources, and (3) tackling the issue of incomplete knowledge in constructed KGs. To achieve these goals, we design a multi-stage integrated procedure, ranging from user segmentation and LLM-supported KG construction process to interconnectedly propagating between the KG and the Interaction Graph (IG). Our experimental results on a telecom e-commerce domain dataset demonstrate that our approach not only makes existing GNN-based recommender baselines feasible on large-scale data but also achieves comparative performance with the HybridGCN core.**

Keywords*—Large-scale dataset processing; recommender systems; graph neural network; knowledge graph construction; data segmentation

I. Introduction

Recommender System (RS) has been playing a pivotal role in enhancing user experience on e-commerce platforms. It uses user historical interactions and item attributes to generate personalized recommendations. Traditional approaches have been developed and can be categorized into two primary pillars: Content-based and Collaborative Filtering. However, they may fall short in practical applications with higher rates of sparsity and cold start [1].

Recently, graph-based modeling has been an emerging trend in the field, as it can exploit and extend the relations between users or items [2]. Graphs provide a natural way to represent and model relationships, capturing complex inter-dependencies and interactions that traditional methods might overlook. Noteworthy, Graph Neural Network-based (GNN-based) techniques have showcased exceptional performance across a myriad of application domains, underscoring the potential and adaptability of this approach. Despite the promising performance, the implementation encounters limitations when applied to large-scale datasets characterized by an extensive volume of users and items, as well as diverse interaction patterns, which can lead to *neighbor explosion* during graph construction [3]. Many recent SOTA GNN-based RS [4], [7], [17] have only experimented on popular benchmark datasets with medium-to-small user bases, questioning their feasibility on real-world systems with large-scale user data.

Incorporating Knowledge Graphs (KGs), which encapsulate domain-specific knowledge and semantic relationships, can further support the recommendation process in the embedding stage [12]. By integrating Graph Neural Networks with Knowledge Graphs, recommender systems can harness both the structural relationships and semantic insights, resulting in more accurate, context-aware, and personalized recommendations [16]. However, a problem lies in the reliance on open sources, which creates challenges in constructing complete Knowledge Graphs for domain-specific private data, thereby limiting the model's applicability and effectiveness in diverse and complex environments. Indeed, recent GNN-KG-combined models [17], [18] have only been evaluated on popular datasets with easily extractable KGs from Open Knowledge Bases, without considering their performance on narrowly specialized domains, thus ignoring issues that may arise in post-synthesized KGs like *incomplete knowledge*.

In this paper, we propose a new recommender system model, **HybridGCN**, which will address all of the above problems. Particularly, our main contributions are as follows:

1) Propose a semi-automatic procedure for constructing our domain-specific knowledge graph in a niche domain that is highly RS-compatible, with support from Large Language Model (LLM).
2) Achieve scalable Graph Convolutional Network (GCN) on empirically large-scale datasets through user behavioral segmentation.
3) Tackle the practical issue of incomplete knowledge integration in GNN-based recommender models leveraging KGs, in which HybridGCN stands as our state-of-the-art (SOTA) approach. We empirically compare HybridGCN with other SOTA methods and demonstrate substantial improvements.

The remainder of this paper is as follows. Section II overviews the related work. Then Section III describes our proposed method, which includes the overall pipeline, model architecture, and training strategy. The experiment evaluation and discussion are detailed in Section IV. Finally, we conclude and discuss our work in Section V.

II. Related Work

Traditional recommender systems primarily rely on two main approaches: collaborative filtering and content-based filtering. Collaborative filtering methods generate recommendations by identifying patterns and similarities among users or items, while the latter recommends items based on their features or attributes, matching user preferences with item characteristics. ALS [20] (Alternating Least Squares) is a popular collaborative filtering algorithm that utilizes matrix factorization to decompose the user-item interaction matrix into lower-dimensional matrices (latent factors) representing users and items similarities on new factors. However, ALS tends to recommend popular items frequently, leading to a lack of diversity and personalization in recommendations.

More advanced techniques have been developed to address those issues, including the use of Neural Networks. One such approach is Mult-VAE [21], which leverages deep learning to build recommender systems. Mult-VAE employs multiple layers of Variational Autoencoders (VAEs), which are generative models capable of learning complex data distributions and capturing underlying patterns in user-item interaction data. Another powerful tool for modeling and analyzing complex relational data, including recommender systems, is Graph Neural Networks (GNNs). NGCF [4] was the first popular GNN model applied to recommender systems, introducing the concept of message passing. This approach enables NGCF to learn enriched representations of users and items by aggregating information from their neighboring nodes in the graph. Inspired by simplified Graph Convolutional Network (GCN) design in SGCN [8], LightGCN [7] only focuses on linearly combining the embeddings obtained from different propagation layers in the graph. Additionally, GraphSAGE [11] offers a more general framework for inductive representation learning on graphs, which has also been adapted for large recommender systems. It operates by sampling and aggregating features from a node's local neighborhood to learn node embeddings that capture the structural properties and relationships. Building upon that, PinSage [19] removes the limitation of storing the entire graph by using random walks to sample graph neighborhoods.

Knowledge Graphs provide a structured representation of information, enabling recommender systems to understand and leverage the semantic context and meaning behind user interactions and item attributes. There have been recent studies applying them to graph-based models, notably KGCN [17]. By integrating domain-specific knowledge and structural insights from Knowledge Graphs, KGCN addresses the limitations of conventional recommendation models and achieves superior performance in capturing user preferences and item characteristics, particularly in complex and diverse recommendation scenarios. However, the diversity and incompleteness of natural knowledge pose practical challenges in customizing the integration process of KGs into graph-based models to effectively take advantage of the provided semantics, while avoiding the introduction of unpredictable noises that can conversely degrade performance [13]. The major difference between our HybridGCN core and the literature is that we will leverage the KG propagation paradigm of KGCN and additionally employ a semantic enrichment mechanism inspired by LightGCN-like methods to utilize subgraphs within the interaction graph for indirectly inferring more hidden connections, which is a *cross-graph propagation technique*.

The construction of a Knowledge Graph is also a challenge. In the context of recommender systems, integrating information from a knowledge graph source with high semantic consistency and low noise is crucial to ensure relevance, and personalization, and enhance the overall quality of recommendations. This also means that each real-world entity should have a unique identifying node within the integrated KG. Typically, knowledge about entities can be collected from various sources, each providing a KG that represents its understanding of the queried entity set. From there, a challenge arises in unifying the different aliases of the same entity that appear in multiple asynchronous data sources [14], [15]. This is accomplished through entity alignment tasks, aiming to create an ultimate comprehensive KG for model learning. For example, KGCN [17] uses an open knowledge base (OKB) to extract item-related triples for constructing their knowledge graphs and testing their model on popular datasets. Due to the nature of OKB, which organizes knowledge in a structured manner through metadata, Resource Description Framework (RDF), or defined ontologies [5], extracting triples and re-connecting them into a knowledge graph input is relatively straightforward and does not heavily rely on the entity alignment step. However, with domain-specific datasets, the construction of semantic triples often requires a more complicated process, involving the extraction and reorganization of information from unstructured data sources [6]. In this paper, we employ an innovative approach using LLMs to capture, denoise, and enrich semantic entities and relations within our Knowledge Graph.

III. Proposed Method

A. Overall Framework

The overall framework of the proposed system is illustrated in Fig. 1(a). The customer base of a commercial system can potentially encompass a large number of individuals, each with diverse needs and usage patterns. Therefore, we propose an initial stage involving segmenting users into distinct groups based on their historical behaviors. Within our context of telecommunication, this segmentation process will leverage side behavior indicators, such as user revenue or historical patterns of calling and data usage.

After the segmentation stage, each user is assigned to a cluster that comprises other users with similar behavior patterns. Personalized recommendations are then generated by considering the items interacted with by users in the same cluster. It is worth noting that in practical scenarios, the system does not possess immediate access to the behavior history of new users. Consequently, for such users, the system offers diverse recommendations based on their initial needs. As the behavior history of new users gradually accumulates, dynamic assignment to existing clusters becomes feasible.

The graph construction stage for each cluster involves transforming user-item interaction data into an interaction graph (IG) and building an item-related KG. Specifically, to construct the IG, we utilize subscription data as an implicit feedback source from users to items. This data type is unary, implying that we can only infer user preferences based on their

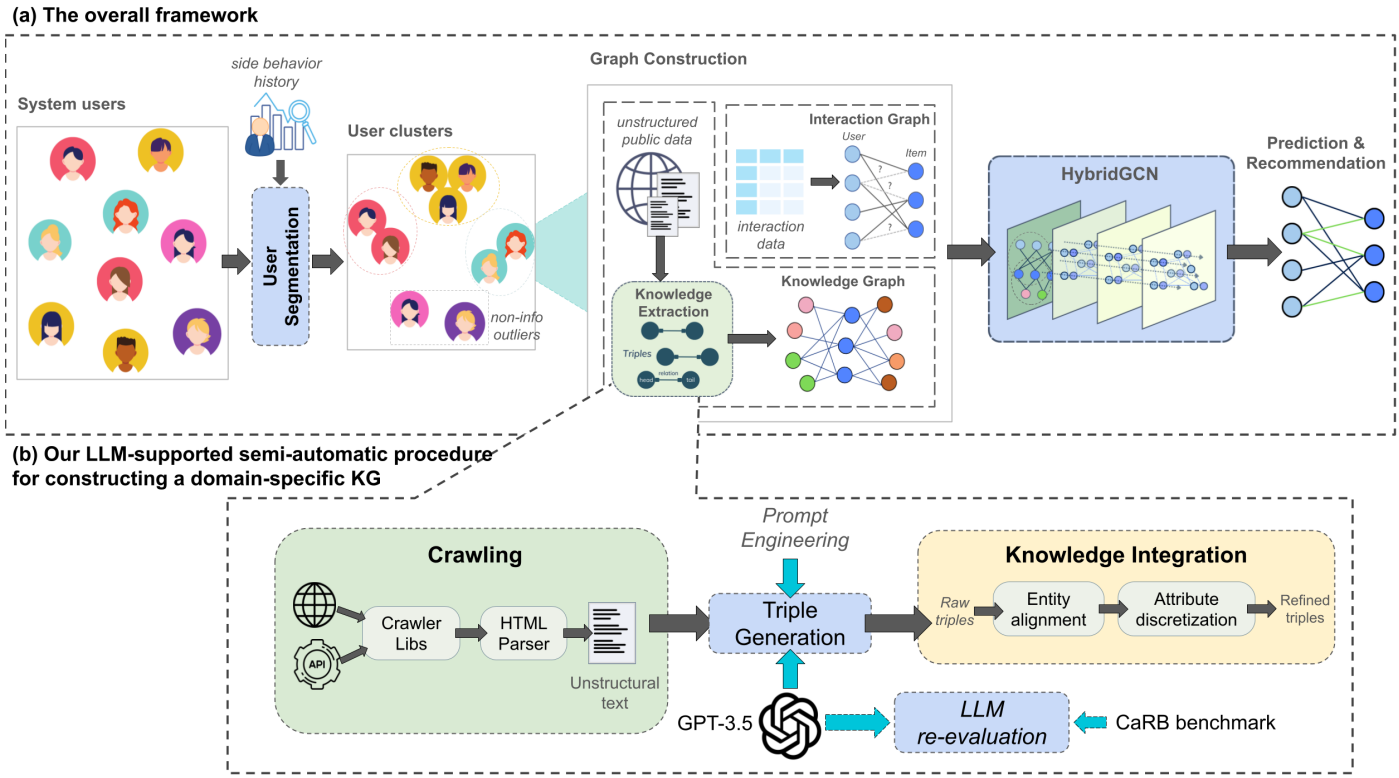


Fig. 1. Overall framework of HybridGCN.

subscribed packages while assigning uncertain probabilities to packages they have not interacted with. As for the item-related KG, we implement an approach supported by a Language Model (LM) to extract data from the internet. The procedure for constructing our domain-specific knowledge graph is illustrated in Fig. 1(b).

In detail, our process consists of three steps, which take place in a semi-automatic manner. Firstly, telecom package descriptions are scraped and parsed from API sources or relevant official websites. The semantic information of these packages includes price, package type, minutes allowed for domestic and international calls, or accompanying benefits. The output of this step is unstructured text corresponding to available packages. It is noted that information about a telecom package may appear in multiple sources. We collect data from various sources to ensure the completeness and diversity of semantic descriptions for these packages. However, there are also some packages for which their external information cannot be found.

In the second step, we utilize the API provided by the GPT-3.5 language model to extract information from the text, returning the results in the form of semantic triples. Several previous studies [23]–[25] have examined the capabilities of generative Language Models in text understanding and generation, demonstrating viable solutions for information extraction tasks. However, when it comes to knowledge extraction tasks, a conversational model that is not specifically trained to recognize entities and relations may not be able to provide an alignable set of entities [26]. To address these challenges, we carry out two following tasks: (1) *specify the prompt engineering by providing the ontology of our domain-specific KG*, such as specifying the types of relations between different entity types; and (2) *pre-train the LM with some labeled domain-oriented data samples*. We also re-evaluate the knowledge extraction capability of GPT-3.5 by using the CaRB benchmark [27] through its information matching and scoring framework on two specific evaluation scenarios. The results from the general scenario on a subset of the TekGen dataset [28] and the scenario on our handcrafted sub-knowledge base (approximately 50 pairs of unstructured text and corresponding extracted triple sets have been human-labeled) are 69.82% and 100% in *Recall*, respectively, indicating the promising performance of GPT-3.5 in this stage's task.

In the final step, we perform the knowledge integration task, which involves entity alignment and discretization for groups of item-related entities with continuous values. It is important to mention that the raw triples obtained from the LM may contain ambiguous values for the same entity, requiring NLP techniques to normalize its identification. Specifically, we need to standardize the units for each numeric value and establish common conventions for descriptions. For example, for the attribute related to minutes allowed for domestic calls in a package, we standardize the unit as *'days'* for packages with daily/weekly cycles and *'months'* for packages with monthly cycles and above, etc. Additionally, for extracted text chunks (entities in raw triples) in natural descriptive language, such as additional package benefits, we remove domain-specific stopwords and cluster them based on their TF-IDF encoded representations. This helps to unify phrases that refer to the same entity but may appear in different text chunks because of misspellings or redundant grammar elements. The final output

of the entire process is a set of refined KG triples that can be integrated into KG-based GNN models.

The indexed graphs serve as the main input for the proposed core model of HybridGCN, whose detailed architecture will be shown in *Fig.* 2.

B. Model Architecture

We combine the idea of graph convolution-based information propagation on the intra-knowledge graph (intra-KG) from KGCN [17] and intra-interaction graph (intra-IG) from LightGCN [4]. Our HybridGCN core adds interconnective paths to create a continuous information propagation flow between the processing components in both types of graphs, aiming to enhance the embedding learning process and address the issue of incompleteness in practical KGs.

First, we note that in embedding spaces, nodes within a graph are represented by finite-dimensional vectors, and the relationship between any two nodes is quantified through operations performed on the corresponding pairs of vectors. Such representation is known as the *ID embedding* of a node.

The concept of intra-KG propagation involves calculating the final representation of a given item entity by incorporating its intra-KG neighborhood information as *neighbor embedding* into its ID embedding via an aggregator (as presented in (Eq.2)). The semantic propagation process is performed on a knowledge graph from the outside to the inside through multiple hops, based on *receptive fields*, which are selected sets of neighboring nodes for each entity node. Through this process, the structural topology of the proximity sub-graph containing an entity node is embedded into the entity itself.

Neighbor embeddings are aggregated using a graph attention mechanism. Considering a node v and its set of neighbor nodes $\mathcal{N}(v)$ at h-th hop, the importance of the relationship between that node and its neighboring nodes is defined based on a weight, which is the normalized inner product $\tilde{\pi}_r^u$ between the ID embedding of the user u linked to the end target item entity and the relation ID embedding r. Therefore, the neighboring information of node v is weight-based linearly combined and then integrated with node v itself to form the resulting embeddings of the h-th hop, which also serve as the input for its adjacent $(h-1)$-th hop. During each hop, user-relation weight π_r^u, normalized user-relation weight $\tilde{\pi}_r^u$, and neighbor embedding $\mathsf{v}_{\mathcal{N}(v)}^u$ are respectively calculated as follows:

$$\begin{aligned} \pi_r^u &= u^T r; \\ \tilde{\pi}_{r_{v,e}}^u &= \frac{exp(\pi_{r_{v,e}}^u)}{\sum_{e\in\mathcal{N}(v)} exp(\pi_{r_{v,e}}^u)}; \\ \mathsf{v}_{\mathcal{N}(v)}^u &= \sum_{e\in\mathcal{N}(v)} \tilde{\pi}_{r_{v,e}}^u e \end{aligned} \tag{1}$$

$$agg = AGG(v, \mathsf{v}_{\mathcal{N}(v)}^u) \tag{2}$$

For each target item node i, we preserve the aggregated neighbor embeddings $\mathcal{V}_{\mathcal{N}(i)}^u$ from the innermost hop, which is formed by combining adjacent nodes of this item node. These final neighbor embeddings are then used as input for our HybridGCN model.

On the other hand, the intra-IG propagation rule is defined based on the user-item connections. The deeper the hops (layers) in the graph neural network, the longer the propagation paths within the graph, such as user-item, user-item-user, item-user-item, etc. Embedding these propagation paths into an ID user (item) embedding helps capture multi-order proximity structure and improve the issue of sparse connections in the graph. Given $\mathcal{N}_i$, $\mathcal{N}_u$ as the set of neighbor nodes of item (user) and $e_i^{(k)}$, $e_u^{(k)}$ as the item (user) ID embeddings at layer k, the graph convolution operation for calculating layer-(k+1) embeddings from layer-k:

$$\begin{aligned} e_u^{(k+1)} &= \sum_{i\in\mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|\cdot|\mathcal{N}_i|}} e_i^{(k)}; \\ e_i^{(k+1)} &= \sum_{u\in\mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_i|\cdot|\mathcal{N}_u|}} e_u^{(k)} \end{aligned} \tag{3}$$

Finally, the embeddings at K layers are weight-combined to form the final representation of a user (an item):

$$\mathbf{e}_u = \sum_{k=0}^{K} \alpha_k e_u^{(k)}; \;\; \mathbf{e}_i = \sum_{k=0}^{K} \alpha_k e_i^{(k)} \tag{4}$$

Our HybridGCN designs facilitate communication between intra-graph propagation components and combine the enriched embeddings from each type of graph to enhance the embeddings in the *inter-graph* context. In *HybridGCNa*, propagation occurs first in the IG space. The propagation also simultaneously takes place in the KG to generate neighbor embeddings for items with equivalent KG entities. The IG-enriched item embedding resulting from the former process and its corresponding neighbor embedding from the latter are then combined using a *sum aggregator* to obtain the final representation of an item. The combination operation in *HybridGCNa* is defined as follows:

$$\mathbf{e}_i^u = SUM\_AGG\left(\sum_{k=0}^{K} \alpha_k e_i^{(k)}, \mathcal{V}_{\mathcal{N}(i)}^u\right) \tag{5}$$

In contrast, *HybridGCNb* allows the semantic propagation on the KG and the combination of an item's initial embedding with its neighbor embedding to occur first (as presented in (Eq. 6)). This results in KG-based pre-enriched item embeddings $\mathbf{e}_i^{u(0)}$, which are then fed into the input embedding matrix to perform the propagation rule on the IG for adopting K representations at K layers $\mathbf{e}_i^{u(1)}, \mathbf{e}_i^{u(2)}, \ldots, \mathbf{e}_i^{u(K)}$, and the final synthesized item embedding $\mathbf{e}_i^u$. Our experimental results show that utilizing a straightforward average aggregation method, instead of relying on user-based weights for item-related relations as the original intra-KG mechanism, simplifies the compilation of neighbor information across our moderate-sized domain-specific KG. Thus, it enhances performance and improves the ease of learning in this version.

$$\mathbf{e}_i^{u(0)} = SUM\_AGG\left(e_i^{(0)}, \mathcal{V}_{\mathcal{N}(i)}^u\right) \tag{6}$$

Our SUM_AGG operation is designed to enable addition between two vectors with different dimensions, based on the

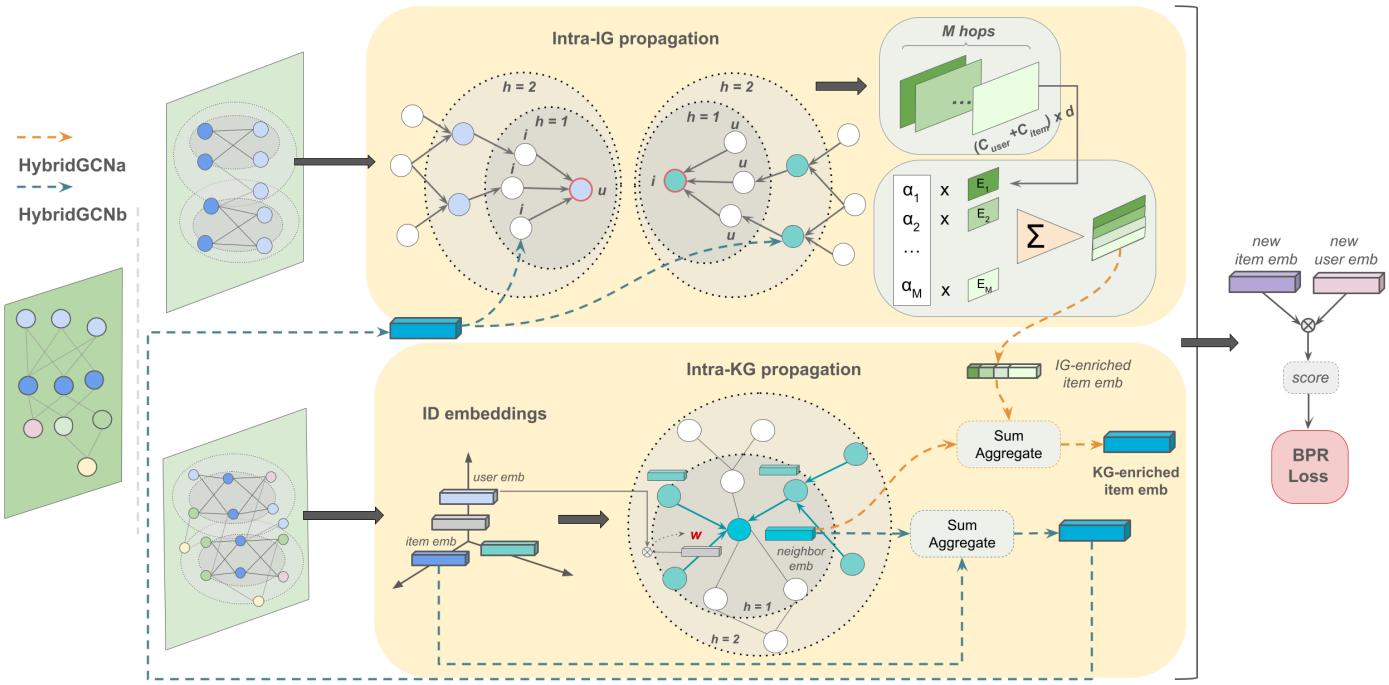


Fig. 2. Detailed architecture of our HybridGCN core model.

expansion of the vector with fewer dimensions on the right side with zero elements. This allows HybridGCN models to flexibly adjust the influence of intra-KG semantic information on item learning.

Finally, the models predict the interaction probability by calculating the inner product between post-propagated representations of user u and item i:

$$\hat{y}_{ui} = \mathbf{e}_u^T \mathbf{e}_i^u \tag{7}$$

To optimize the performance of our model, we utilize the *Bayesian Personalized Ranking* (BPR) loss [29], which is also employed by LightGCN. This loss function aims to ensure that the predicted value for an observed item is higher than the predicted values for unobserved items:

$$\begin{aligned} L_{BPR} = -\sum_{u=1}^{M} \sum_{i \in \mathcal{N}_u} \sum_{j \notin \mathcal{N}_u} ln\, \sigma(\hat{y}_{ui} - \hat{y}_{uj}) \\ +\lambda_1 \|\mathbf{E}^{(0)}\|^2 + \lambda_2 (\|\mathbf{R}\|^2 + \|\mathbf{A}\|^2) \end{aligned} \tag{8}$$

where λ_1, λ_2 controls the L_2 regularization strength for the user-item embedding matrix, the batch's existing inner-KG attribute embedding matrix, and the relation embedding matrix. Our optimization process utilizes the Adam optimizer in a mini-batch fashion.

C. Model Analysis

We perform mathematical analysis to illustrate the reasoning behind the inter-graph design of HybridGCN. Initially, we provide a theoretical discussion on how HybridGCN can tackle the issue of unavailability of knowledge when integrating real-world Knowledge Graphs. Subsequently, we highlight the significance of learning the interconnections between two graph types in enriching the semantics of inherently sparse interaction data.

1) Alleviation of knowledge incompleteness: In practice, there exist items that do not have corresponding entities in the constructed Knowledge Graphs due to unavailability of information, referred to as *isolated items*. This asymmetry in information gives rise to bias or unexpected noise when relying solely on KG-extracted semantics for learning item embeddings. The reason is that there is uncertainty regarding whether an isolated item in reality shares certain characteristics with known item nodes.

Our inter-graph propagation in the HybridGCNb setting helps address this incompleteness by inferring hidden relationships between isolated items and existing attribute-related entities on the KG. As depicted in Fig. 3, through interconnected propagation on both the IG and KG, the embeddings of attributes a_1 and a_2 are integrated into u_1, and then the u_1 embedding is propagated to i_3 *(an isolated item)* via the pair connection $u_1 - i_3$, thereby intuitively forming an indirect connection $i_3 - a_1$, and $i_3 - a_2$.

To clarify, with the integration of intra-KG neighbor information into the initial item embedding before propagating it on the IG, we can expand the representation of an item in the

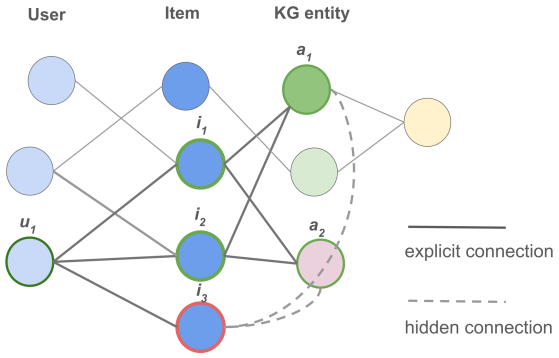


Fig. 3. Inter-graph propagation in IG and KG facilitates inferring unknown connections between *isolated item* and attribute-related entities.

second layer of IG-based graph convolution as follows:

$$\begin{aligned}
\mathbf{e}_i^{u(2)} &\overset{(3)}{=} \sum_{u\in\mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_i|\cdot|\mathcal{N}_u|}} e_u^{(1)} \\
&\overset{(3)}{=} \sum_{u\in\mathcal{N}_i} \frac{1}{|\mathcal{N}_u|} \sum_{j\in\mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_i|\cdot|\mathcal{N}_j|}} \mathbf{e}_j^{u(0)} \\
&\overset{(6)}{=} \sum_{u\in\mathcal{N}_i} \frac{1}{|\mathcal{N}_u|} \sum_{j\in\mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_i|\cdot|\mathcal{N}_j|}} \left(e_j^{(0)} + \mathcal{V}_{\mathcal{N}(j)}^u \right) \\
&\overset{(1)}{=} \sum_{u\in\mathcal{N}_i} \frac{1}{|\mathcal{N}_u|} \sum_{j\in\mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_i|\cdot|\mathcal{N}_j|}} \left(e_j^{(0)} + \sum_{\substack{e\in \\ \mathcal{N}^{KG}(j)}} \tilde{\pi}_{r_{j,e}}^u e \right)
\end{aligned} \tag{9}$$

Considering (Eq. 9), we observe that in the case where item i and item j both receive interactions from a user or a group of users, the second-layer representation of i is proportional to the KG-extracted neighbor embeddings of j through a coefficient:

$$c_{j,i} = \frac{1}{\sqrt{|\mathcal{N}_i|\cdot|\mathcal{N}_j|}} \sum_{u\in\mathcal{N}_i\cap\mathcal{N}_j} \frac{1}{|\mathcal{N}_u|} \tag{10}$$

The aforementioned hidden connections (as illustrated in Fig. 3) are established based on this coefficient, particularly when item j is an isolated item. Following (Eq. 10), the strength of these relationships is determined by:

(1) The greater the number of users jointly interacting with both items i and j, the stronger these hidden connections are. This is reasonable because a dense collaboration of users regarding the pair (i, j) indicates a higher likelihood of these items sharing similar characteristics.
(2) The less popular item i and item j are, the larger the magnitude tends to be. This also implies the group that items i and j belong to exhibits a high degree of personalization.
(3) User interaction engagement level is also considered. A lower level of item interaction corresponds to a higher level of confidence in hidden relationships' existence.

In terms of the HybridGCNa setting, according to (Eq. 5), the final item representation is a straightforward combination of the results obtained from two propagation processes: intra-IG and intra-KG. Specifically, it encompasses the IG-enriched item embeddings and the KG-based neighbor embedding. As a result, compared to KGCN, HybridGCNa can balance and regularize semantic learning, moderating the over-dependence on noisy knowledge triples and mitigating the asymmetry in the availability of information across practical KGs.

2) Augmentation of sparse interaction data: In many recommendation scenarios, models may face sparse user-item collaborative data or cold-start issues with new items that have limited interactions from users, as well as a few highly specialized items. This causes challenges for multi-level propagation based solely on graph convolution within the interaction graph, as employed by LightGCN. Naturally, semantic structures extracted from the knowledge graphs can be integrated into collaborative information to provide more detailed and specialized representations for items. This resembles the paradigm of traditional hybrid recommendation systems but within the context of state-of-the-art graph-based models. Eq. (5) of HybridGCNa once again provides a direct observation of this combination, wherein external knowledge complements user-item interaction data.

Regarding HybridGCNb, some transformations are needed to observe how semantically rich connections from knowledge graphs augment and enrich user-item collaborative data. Based on (Eq. 4), (6) and (9), we can unfold the final embedding of an item in the HybridGCNb setting as follows:

$$\begin{aligned}
\mathbf{e}_i^u &\overset{(4)}{=} \sum_{k=0}^{K} \alpha_k \mathbf{e}_i^{u(k)} \\
&= \alpha_0 \mathbf{e}_i^{u(0)} + \alpha_1 \mathbf{e}_i^{u(1)} + \alpha_2 \mathbf{e}_i^{u(2)} + \dots \\
&\overset{(6)(9)}{=} \alpha_0 \left(e_i^{(0)} + \mathcal{V}_{\mathcal{N}(i)}^u \right) + \alpha_1 \mathbf{e}_i^{u(1)} \\
&\quad + \alpha_2 \left(\sum_{u\in\mathcal{N}_i} \frac{1}{|\mathcal{N}_u|} \sum_{\substack{j\in\mathcal{N}_u \\ j\neq i}} \frac{1}{\sqrt{|\mathcal{N}_i|\cdot|\mathcal{N}_j|}} \left(e_j^{(0)} + \mathcal{V}_{\mathcal{N}(j)}^u \right) \right. \\
&\quad \left. + \frac{1}{|\mathcal{N}_i|} \sum_{u\in\mathcal{N}_i} \frac{1}{|\mathcal{N}_u|} \left(e_i^{(0)} + \mathcal{V}_{\mathcal{N}(i)}^u \right) \right) + \dots \\
&= \left(\alpha_0 e_i^{(0)} + \alpha_1 e_i^{(1)} + \alpha_2 e_i^{(2)} + \dots \right) \\
&\quad + \left(\left(\alpha_0 + \alpha_2 \frac{1}{|\mathcal{N}_i|} \sum_{u\in\mathcal{N}_i} \frac{1}{|\mathcal{N}_u|} + \dots \right) \mathcal{V}_{\mathcal{N}(i)}^u \right. \\
&\quad \left. + \alpha_2 \frac{1}{|\mathcal{N}_i|} \sum_{u\in\mathcal{N}_i} \frac{1}{|\mathcal{N}_u|} \sum_{\substack{j\in\mathcal{N}_u \\ j\neq i}} \frac{1}{\sqrt{|\mathcal{N}_j|}} \mathcal{V}_{\mathcal{N}(j)}^u + \dots \right)
\end{aligned} \tag{11}$$

It is noted that $\mathbf{e}_i^{u(2k+1)} = e_i^{(2k+1)}$ because the results of graph convolution at odd layers can be easily unfolded all the way to initial user embeddings (0-layer user representations).

Based on the expansion in (Eq. 11), we can observe that the ultimate item embedding has been enriched with additional blocks of external semantic information. In this way, not only the structural topology of an item's intra-KG neighborhood

(denoted as $\mathcal{V}^u_{\mathcal{N}(i)}$) is encapsulated, but also the neighbor subgraph encodings of other items (denoted as $\mathcal{V}^u_{\mathcal{N}(j)}$ with $j \neq i$) are included and indirectly related to the target item through interaction data (explained in Section III-C1). Additionally, such subgraphs are integrated with different levels of smoothness, which are adjusted based on the size of the neighboring region within the learned graphs.

IV. Experimental Results

A. Dataset

In this study, we applied our proposed framework to analyze behavioral data obtained from a prominent telecommunications service provider. This data encompasses user activity logs spanning three months, from November 2022 to January 2023. It includes anonymized information on user package subscriptions, actual usage logs, and package metadata (see Table I for dataset summary). To ensure user privacy, all sensitive data has been encrypted.

TABLE I. Summary of the Dataset's Key Characteristics

Characteristic	*Value*
Number of unique users (Subscription behavior)	10,630,045
Number of unique users (Usage behavior)	5,065,934
Number of packages	2,283
Sparsity	0.9986

B. Baselines

Our baseline models encompass a diverse range of approaches, including traditional, state-of-the-art, and graph-based models.

- **SVD** [9]: A classic CF-based model that uses inner product operations to represent user-item interactions.
- **SVD++** [9]: SVD++ enhances its original version of SVD by incorporating implicit feedback inferred from user behaviors. We utilize the implementations of SVD and SVD++ provided by Surprise library [10].
- **ALS** [20]: ALS is a matrix factorization method that is common in many real-world recommendation scenarios. ALS decomposes the original utility matrix into two matrices by iteratively updating the values of the user and item latent factor matrices, which is achieved by solving a least squares problem at each iteration. We use its implicit version in PySpark.
- **Mult-VAE** [21]: Mult-VAE is a deep learning model that leverages variational inference to learn latent representations of user-item interactions.
- **PageRank**: A Random Walk-inspired graph learning technique ranks items based on their graph-based importance, considering the global structure of the user-item interaction network. It then generates top-ranked recommendations for the N most important products universally across all users. We implement it based on the NetworkX library [22].
- **LightGCN** [7]: LightGCN is a lightweight graph-based model, which simplifies the graph convolution operation to focus solely on multi-hop user-item interactions, making it efficient and effective for learning from large-scale recommendation datasets.
- **KGCN** [17]: KGCN is a graph convolutional network-based model that captures neighborhood structures within knowledge graphs to improve personalized recommendation capabilities.

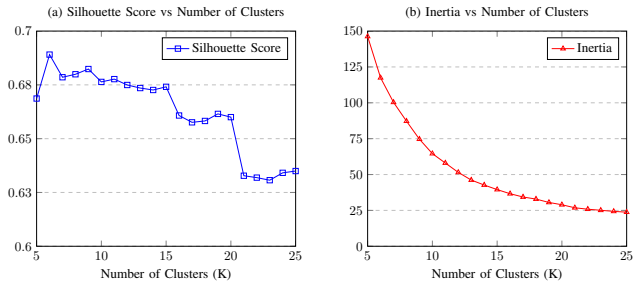


Fig. 4. Evaluation of KMeans Clustering Algorithm with *(a) Sihouette Score* and *(b) Inertia*

C. Experimental Setup

a) Data Preprocessing: We removed any rows with NULL or NaN values, which only occur at a mere 1% of the dataset. Subsequently, a pivoting operation was performed for each user, yielding two distinct pivoted datasets: *Dataset 1* for all the packages that each user subscribes to, *Dataset 2* for his/her historical usage behavior.

b) Feature Engineering: KMeans clustering was conducted on *Dataset 2*. Based on cluster evaluations, an optimal value of **K = 20** was determined (see Fig. 4 for more detail). Users who have subscribed but have not recorded any behavior in *Dataset 1* are categorized into a separate *Cluster -1*. Users subscribing to only one package were also excluded from the analysis for a more rigorous evaluation, and the IQR method was then employed to remove outliers. The *Final Dataset* comprises entries for both the subscribed packages and their respective cluster indices. In Fig. 5, we show the historical revenue from users for each cluster in *December 2022*, highlighting the differences in user behavior across the clusters. Furthermore, from the knowledge graph constructed across all possible packages, we extracted the subgraph corresponding to each cluster. In Table II, we evaluated the proportion of packages that have been interacted with by users belonging to the cluster and have corresponding entities in the cluster's KG, defined as *Hit rate*. This rate reflects the incompleteness of KG's understanding of entities representing packages.

TABLE II. Statistics on the Proportion of Telecom Packages Having Corresponding Entities in the KG (a.k.a Hit Rate) for Each User Cluster (%)

Cluster	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	-1
Hit rate	71	70	76	100	86	69	69	76	71	80	73	90	73	70	69	71	72	71	*n/a*	72	68

c) Train-Test Split: To ensure fairness for all models, the data trained and tested must be the same. Furthermore, we have also devised a strategy to capture personalization and cold-start solving capability.

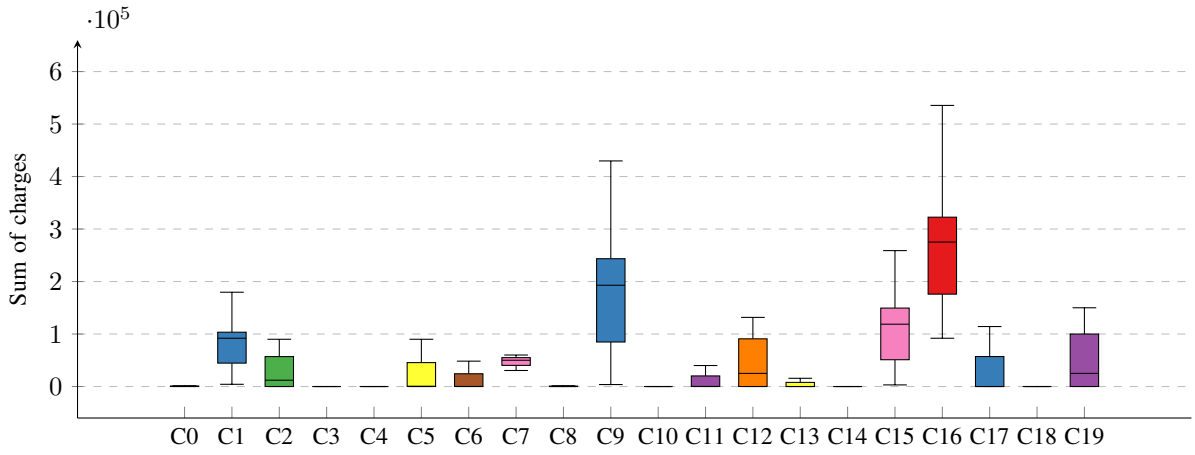


Fig. 5. Boxplot representing historical revenue from users (a.k.a their sum of charges) in december 2022 across clusters in the final dataset.

- We divide the *Final Dataset* into two user groups: those who are supposed to arrive earlier and those who are supposed to arrive later, in a $1:1$ ratio.
- We randomly hide some *"less popular"* packages of each user in the latter group to create a *test set* (corresponding to the darker cells in Fig. 6). The remaining user-item interactions (corresponding to the white cells in Fig. 6) will form the *training set*. This approach creates a realistic asymmetric scenario where the model is forced to predict unpopular items. When evaluating, we consider the model's predictions for hidden packages of all users in the latter group. More specifically, we regard a package as *"less popular"* if it does not belong to the top two most subscribed packages. Based on our observations from statistics, these two packages are interacted with by almost all users. This implies that they might be free packages given periodically by the service provider, so including them in the evaluation does not provide much meaningful insight.

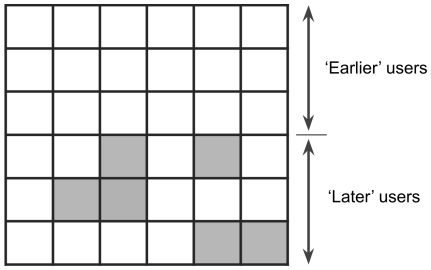


Fig. 6. Half of the users belong to the group of 'earlier' users, where all the packages they have subscribed to are included in *training set*. The rest belong to the group of 'later' users, where some of their *'less popular'* packages (represented as darker cells) are randomly selected to form *test set*.

d) Hyperparameter Settings: Hyperparameters among intra-cluster models in all baselines are set the same to ensure fairness in the evaluation process. Regarding two versions of HybridGCN and KGCN models, the intra-KG neighbor sampling size is set between 2-4, depending on the size of the cluster's knowledge graph (KG), and the depth of the receptive field is set to 1 due to the relatively low complexity of our domain-specific KG. In HybridGCN and LightGCN, the number of intra-IG layers is set to 1-5, with a larger number of layers chosen when the number of users in the cluster increases, enabling the learning of longer propagation paths. The layer weights in HybridGCN and LightGCN are uniformly set as $\alpha_0 = \alpha_1 = \alpha_2 = \ldots = \alpha_K = \frac{1}{K+1}$, following the configuration specified in the LightGCN paper. The regularization coefficients in HybridGCN's loss function are set as $\lambda_1 = \lambda_2 = 10^{-4}$, which are also equivalent to the corresponding hyperparameter λ chosen in LightGCN.

D. Results

In this paper, the results were obtained using standalone Colab Pro[1] (51GB RAM, NVIDIA A100 GPU). We evaluate our approach through top-K recommendation, where trained models predict the probability of user-item interactions to select K items with the highest scores for each user in the test set. We employ a rank-based metric set for model evaluation. Due to the significantly smaller number of packages (items) compared to the number of users, we find that using $K = 5$ and $K = 10$ provides a sufficiently objective assessment in our study.

- *Precision (P@K)* measures how many items with the top K positions are relevant.
- *Recall (Recall@K)* measures the share of relevant items captured within the top K positions.
- *Mean Reciprocal Rank (MRR@K)* quantifies the rank of the first relevant item found in the recommendation list.
- *Normalized Discounted Cumulative Gain (nDCG@K)* focuses on the relevant item's position in search results. It assigns higher scores to items that are ranked higher and gradually decreases the score as the position decreases.

[1]Google Colab is a cloud-based service provided by Google that allows users to write, execute, and share Python code in a web-based Jupyter Notebook interface.

- *Mean Average Precision (mAP@K)* averages the P@K metric at each relevant item position in the recommendation list.

In both scenarios, whether integrated with clustering or not, both SVD and SVD++ exhibit poor recommendation performance due to the extremely high sparsity of collaborative data. Without clustering, GNN-based baseline models (namely KGCN and LightGCN) cannot perform as large-scale graphs use much more memory than other methods for storing the interaction matrix and embedding spaces, while Implicit ALS is the best possible model in this scenario. Mult-VAE performs fairly well in recall but shows lower ranking-quality scores compared to ALS. This is particularly evident in the case of PageRank, as it solely relies on item popularity and disregards their ranking (as presented in Table III).

TABLE III. COMPARISON OF RECOMMENDATION MODELS WITHOUT CLUSTERS ON OUR DATASET (%)

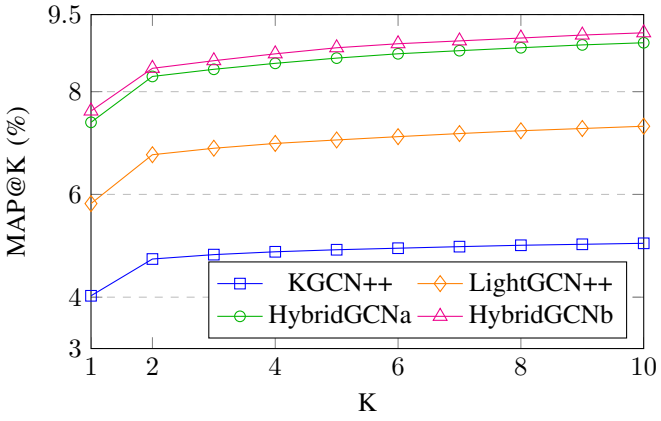
Method	K = 5					K = 10				
	P	*Recall*	*MRR*	*nDCG*	*mAP*	*P*	*Recall*	*MRR*	*nDCG*	*mAP*
SVD	0.11	0.50	0.38	0.37	0.27	0.25	1.96	0.61	0.87	0.27
SVD++	0.11	0.44	0.37	0.36	0.27	0.22	1.76	0.58	0.80	0.27
ALS	**4.14**	**16.45**	**10.38**	**11.03**	**4.58**	3.25	25.87	**11.76**	**14.23**	**4.70**
Mult-VAE	3.61	14.58	7.00	8.40	1.61	**3.66**	**29.05**	9.17	13.38	1.70
PageRank	3.93	14.98	5.10	7.22	0.09	3.57	28.01	7.03	11.68	0.11

In intra-cluster predictions, the models marginally exhibit higher performance. Notably, incorporating the user segmentation stage into available GNN-based methodologies makes it feasible in the setting of limited memory resources. Their corresponding clustering-driven versions, named KGCN++ and LightGCN++, demonstrate superior performance compared to non-GNN-based approaches by a significant margin.

Our HybridGCN models achieve the highest level of effectiveness across all metrics on this real-world dataset, with HybridGCNa and HybridGCNb performing best intermittently. In particular, HybridGCN significantly improves ranking quality metrics (MRR, nDCG, and mAP) to a noteworthy extent. It surpasses clustering-driven state-of-the-art models such as LightGCN++ by approximately 1-2%, and KGCN++ by around 4-9% as shown in Table IV and Fig. 7. This highlights the strong capability of our proposed model in addressing the challenge of knowledge incompleteness in KG-based GNN models such as KGCN (see Table II). Moreover, the improvement over LightGCN++ demonstrates that additionally incorporating semantic structures through intra-KG propagation enhances the personalization capabilities of graph learning-based systems. Similar to traditional hybrid approaches, our inter-graph propagation also aids in mitigating the potential issue of sparse collaborative data and cold-start problems.

Between the two variants of HybridGCN, HybridGCNa performs slightly better in overall evaluation metrics such as Precision and Recall, while HybridGCNb generally shows a slight advantage in fine-grained ranking quality evaluation metrics like MRR and mAP. These results indicate that HybridGCNa has better generalization ability, while HybridGCNb excels in providing detailed and personalized recommendations based on user preferences. This is reasonable considering the mathematical interpretations of these two versions in Section III-C. While the formula of HybridGCNa is a straightforward combination of intra-KG and intra-IG propagation results, HybridGCNb enables a smoother mechanism when integrating different semantic structures from KGs into collaborative data.

(a) The results of *MAP@K*



(b) The results of *Recall@K*

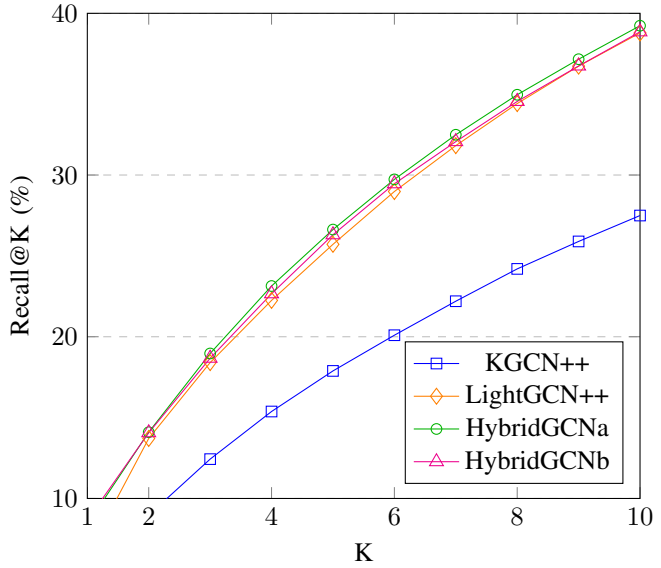


Fig. 7. Detailed evaluation of graph neural network-based models in top-K recommendation with *(a) MAP@K* and *(b) Recall@K*.

TABLE IV. COMPARISON OF RECOMMENDATION MODELS WITH CLUSTERS ON OUR DATASET (%)

Method	K = 5					K = 10				
	P	*Recall*	*MRR*	*nDCG*	*mAP*	*P*	*Recall*	*MRR*	*nDCG*	*mAP*
SVD	0.52	1.98	0.99	1.13	0.32	0.68	5.32	1.53	2.27	0.33
SVD++	0.54	2.06	1.03	1.18	0.33	0.70	5.43	1.58	2.33	0.33
ALS	3.92	15.54	9.44	10.16	3.91	3.34	26.60	11.05	13.90	4.04
Mult-VAE	3.05	11.92	6.60	7.37	2.43	3.41	27.11	8.83	12.48	2.56
PageRank	3.96	15.08	4.97	7.15	0.04	3.68	28.96	6.98	11.85	0.04
KGCN++	4.53	17.89	11.14	11.98	4.92	3.49	27.49	12.56	15.26	5.05
LightGCN++	6.39	25.71	15.95	17.29	7.06	4.84	38.78	17.79	21.74	7.33
HybridGCNa	**6.61**	**26.63**	17.40	**18.49**	8.65	**4.89**	**39.23**	19.18	**22.79**	8.95
HybridGCNb	6.53	26.31	**17.45**	18.42	**8.85**	4.84	38.85	**19.22**	22.69	**9.15**

V. Discussion

Previous experiments on GNN-based RS [4], [7], [17] have predominantly focused on comparing the effectiveness of models on popular benchmark datasets, where user bases are relatively small, and the setup, such as building KGs as their input, is relatively straightforward. In contrast, we aim to evaluate the feasibility and applicability of deploying such graph-based approaches on a real-world, large-scale dataset where all relevant practical issues need to be considered. Our experiments cover a wide range of evaluations, from global recommendations to recommendations within specific user clusters. We compare some existing efficient methods, examining whether simpler methods can outperform more complex ones in real-life scenarios. We also compare modern GNN-based methods that incorporate knowledge graphs with those that do not. Through experiments, our proposed method has shown better results, taking advantage of hidden information from data based on the graphs we have built. However, to further ensure the practical capacity of our method across various domains, more experiments need to be conducted on datasets from different fields, where KG construction and user behavior can vary.

VI. Conclusion

We propose a comprehensive approach leveraging Knowledge Graphs (KGs) and Graph Neural Networks (GNNs) to address graph-based recommender system problems. Through clustering, our framework shows the feasibility of applying the GNN paradigm to large-scale data. Combining two innovative graph learning structures, our core HybridGCN model adopts a GNN-based technique on cross-graph propagation effectively. It overcomes the limitations inherent in each approach by effectively handling knowledge incompleteness within practical Knowledge Graphs and addressing the sparse connection density in Interaction Graphs. Furthermore, we successfully tackle the challenge of constructing a Knowledge Graph from domain-specific unstructured data by harnessing the capabilities of LLMs, resulting in competitively high Knowledge Graph completion rates across different clusters. We evaluate our approach on a real-world telecommunications dataset using a rigorous assessment strategy. Our methodology successfully applies GNN-based methods to a dataset with millions of users. Specifically, for ranking-centric scores, HybridGCN has demonstrated its effectiveness in personalized recommendation tasks, outperforming other GNN-based models and state-of-the-art methods.

Acknowledgment

The authors acknowledge Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for supporting this study.

References

[1] F. Zhu, Y. Wang, C. Chen, J. Zhou, L. Li and G. Liu, "Cross-Domain Recommendation: Challenges, Progress, and Prospects," Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI), pp. 4721-4728, 2021.

[2] S. Wang, L. Hu, Y. Wang, X. He, Q. Z. Sheng, M. A. Orgun, L. Cao, F. Ricci and P. S. Yu, "Graph Learning based Recommender Systems: A Review," Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI), pp. 4644-4652, 2021.

[3] X. Gao, W. Zhang, J. Yu, Y. Shao, Q. V. H. Nguyen, B. Cui and H. Yin, "Accelerating Scalable Graph Neural Network Inference with Node-Adaptive Propagation," Proceedings of the 2024 IEEE 40th International Conference on Data Engineering (ICDE), 2024.

[4] X. Wang, X. He, M. Wang, F. Feng and T. Chua, "Neural Graph Collaborative Filtering," Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19), pp. 165-174, 2019.

[5] J. Chicaiza and P. Valdiviezo-Diaz, "A Comprehensive Survey of Knowledge Graph-Based Recommender Systems: Technologies, Development, and Contributions," Information 2021, vol. 12(6):232, https://doi.org/10.3390/info12060232, 2021.

[6] G. Agrawal, Y. Deng, J. Park, H. Liu and Y-C. Chen, "Building Knowledge Graphs from Unstructured Texts: Applications and Impact Analyses in Cybersecurity Education," Information 2022, vol. 13(11):526, https://doi.org/10.3390/info13110526, 2022.

[7] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang and M. Wang, "LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation," Proceedings of SIGIR'20, pp. 639–648, 2020.

[8] F. Wu, T. Zhang, et al., Simplifying Graph Convolutional Networks," Proceedings of the 36th International Conference on Machine Learning, 2019.

[9] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," Proceedings of KDD'08, pp. 426–434, 2008.

[10] N. Hug, "Surprise: A Python library for recommender systems," Journal of Open Source Software, vol. 5(52), pp. 2174, 2020. Available: https://doi.org/10.21105/joss.02174

[11] W. L. Hamilton, R. Ying and J. Leskovec, "Inductive Representation Learning on Large Graphs," Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17), pp. 1025–1035, 2017.

[12] B. Abu-Salih, "Domain-specific knowledge graphs: A survey," Journal of Network and Computer Applications, vol. 185(5):103076, 1 July 2021.

[13] H. Yang, Z. Lin and M. Zhang, "Rethinking Knowledge Graph Evaluation Under the Open-World Assumption," Advances in Neural Information Processing Systems (NeurIPS-22), 2022.

[14] D. Chaurasiya, A. Surisetty, N. Kumar, A. Singh, V. Dey, A. Malhotra, G. Dhama, and A. Arora, "Entity Alignment For Knowledge Graphs: Progress, Challenges, and Empirical Studies," *arXiv preprint arXiv:2205.08777*, 2022.

[15] B. D. Trisedya, J. Qi, R. Zhang, "Entity Alignment between Knowledge Graphs Using Attribute Embeddings," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33(01), pp. 297-304, 2019.

[16] F. Sola, D. Ayala, I. Hernández and D. Ruiz, "Deep embeddings and Graph Neural Networks: using context to improve domain-independent predictions," The 53th International Journal of Research on Intelligent Systems, vol. 53, pp. 22415–22428, 2023.

[17] H. Wang, M. Zhao, X. Xie, W. Li and M. Guo, "Knowledge Graph Convolutional Networks for Recommender Systems," Proceedings of the 2019 World Wide Web Conference (WWW'19), pp. 3307–3313, 2019.

[18] H. Wang, F. Zhang, et al., "Knowledge-aware Graph Neural Networks with Label Smoothness Regularization for Recommender Systems," Proceedings of the 25th ACM SIGKDD, pp. 968–977, 2019.

[19] R. Ying, R. He, "Graph Convolutional Neural Networks for Web-Scale Recommender Systems," Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'18), 2018.

[20] Y. Koren, R. Bell and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," Computer, vol. 42(8), pp. 30–37, 2009.

[21] D. Liang, R. G. Krishnan, M. D. Hoffman and T. Jebara, "Variational Autoencoders for Collaborative Filtering," Proceedings of the 2018 World Wide Web Conference (WWW'18), pp. 689–698, 2018.

[22] A. A. Hagberg, D. A. Schult and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX", 2008.

[23] D. Xu, W. Chen, W. Peng, C. Zhang, T. Xu, X. Zhao, X. Wu, Y. Zheng and E. Chen, "Large Language Models for Generative Information Extraction: A Survey," *arXiv preprint arXiv:2312.17617*, 2023.

[24] D. Reichenpfader, H. Müller and K. Denecke, "Large language model-

based information extraction from free-text radiology reports: a scoping review protocol," BMJ open, vol. 13(12):e076865, 9 December 2023.

[25] H. Wu, Y. Yuan, L. Mikaelyan, A. Meulemans, X. Liu, J. Hensman and B. Mitra, "Structured Entity Extraction Using Large Language Models," *arXiv preprint arXiv:2402.04437*, 2024.

[26] M. Trajanoska, R. Stojanov and D. Trajanov. "Enhancing Knowledge Graph Construction Using Large Language Models," *arXiv preprint arXiv:2305.04676*, 2023.

[27] S. Bhardwaj, S. Aggarwal and Mausam, "CaRB: A Crowdsourced Benchmark for Open IE," Proceedings of EMNLP-IJCNLP'19, pp. 6262–6267, 2019.

[28] O. Agarwal, H. Ge, S. Shakeri and R. Al-Rfou, "Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training," Proceedings of NAACL'21, pp. 3554–3565, 2021.

[29] S. Rendle, C. Freudenthaler, Z. Gantner and L. Schmidt-Thieme, "BPR: Bayesian Personalized Ranking from Implicit Feedback," Proceedings of UAI2009, pp. 452-461, 2009.