universität innsbruck

Sequential Recommendation

A Graph-based Perspective

Andreas PEINTNER, 01515339

Innsbruck, August 2025

Dissertation

eingereicht an der Universität Innsbruck, Fakultät für Mathematik, Informatik und Physik zur Erlangung des akademischen Grades

Doctor of Philosophy

Doctor of Philosophy – Doktoratsstudium Informatik

Hauptbetreuerin: assoz. Prof. Dr. Eva Zangerle
Institut für Informatik

Betreuer: Prof. Dr. Günther Specht
Institut für Informatik

Fakultät für Mathematik, Informatik und Physik

... to my family.

Abstract

Sequential recommendation models are systems that predict a user's next action by leveraging the temporal order of their past interactions. Due to challenges like data sparsity, noise, and evolving user interests, accurately modeling these sequences remains an unsolved problem despite efforts using methods from Markov chains to recurrent neural networks. The emergence of Graph Neural Networks (GNNs) prompted a new wave of powerful models as researchers are pushed to capture more complex item relationships.

This thesis contributes a series of new graph-based approaches that directly tackle central challenges in sequential recommendation. We design methods that integrate item features into graph representations, improve efficiency and robustness under sparse and noisy data, and capture both short-term dynamics and long-term repeated user intents through advanced temporal modeling. In addition, we demonstrate the broader applicability of these ideas in related domains such as music emotion recognition. Collectively, these contributions establish new modeling principles that advance the accuracy, efficiency, and interpretability of sequential recommendation systems.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisor, Eva Zangerle. Her guidance, unwavering support, and insightful feedback have been the cornerstone of this dissertation. Eva, your mentorship has not only shaped my research but has also profoundly influenced my development as a scientist. Thank you for providing me with the freedom to explore my own ideas while always being there to steer me in the right direction. I am also immensely grateful to my second supervisor, Günther Specht, for his valuable perspectives and constructive criticism throughout this journey.

My sincere thanks go to my colleagues at the Databases and Information Systems (DBIS) group. The collaborative and stimulating environment you all fostered made the challenges of a PhD much more manageable. The countless discussions, brainstorming sessions, and lunch breaks were invaluable, and I am grateful for the camaraderie and support.

I could not have completed this journey without the endless love and encouragement of my family. To my parents and my brothers, thank you for your constant belief in me and for your unwavering support through all the ups and downs. Your encouragement has been a source of strength. To my parents especially, I am deeply grateful for the guidance and steady support you have given me, which shaped every step of this journey.

Finally, a special and heartfelt thank you to my girlfriend, Sarah. Your love, patience, and understanding have been my anchor. Thank you for being by my side, for celebrating the successes, and for providing comfort during the difficult times.

Contents

Part I.

Preface

1. Introduction

Personalization through recommender systems (RS) is a fundamental component of online platforms, playing a key role in enhancing user satisfaction and engagement. These systems aim to predict a user's preference for items and improve the overall platform experience by suggesting personalized content. Traditional recommendation approaches can be broadly categorized into collaborative filtering and content-based filtering methods [38].

Collaborative filtering predicts user preferences based on the interests of similar users. For example, if users A and B share interests in several items, it's likely they will also share preferences for other items [48, 51]. In contrast, content-based methods rely solely on a user's historical positive interactions to suggest similar items. For instance, if a user frequently listens to a specific singer, a content-based system would likely recommend other songs by the same artist [12, 39, 60]. These traditional approaches typically model user-item interactions in a static fashion, ignoring the temporal structure of the interaction history—such as the order of events or their timestamps. As a result, they primarily capture a user's general preference, rather than their current interest or evolving behavior [65].

To address this limitation, sequential recommender systems (SRS) have emerged. Unlike static models, SRS explicitly account for the temporal order of user interactions, modeling how preferences evolve over time. These systems aim to recommend the next likely item or a sequence of items, by capturing both short-term and long-term dependencies in user behavior [65]. This allows them to better reflect a user's current intent, leading to more accurate and personalized recommendations.

Using SRS for recommendations has distinct advantages over general recommender systems. In real-world scenarios, interactions mostly happen successively and are not isolated from each other. Figure 1.1 shows an example of a shopping spree of User A. In

Figure 1.1.: An example of SR: User A booked a flight, a hotel and rented a car. What will be their next action?

this scenario (the user is booking a holiday), each action depends on the prior ones and so all interactions are sequentially dependent: As a next action User A might book tickets for a tourist attraction. This example also shows that user-item interactions usually happen in a certain sequential context. Additionally, the preference of the user and the popularity of different items (e. g., music or clothing) are dynamic rather than static over time due to personal development and trends [46]. These typical characteristics of online interaction sequences are captured by SRS, but are hard to model with traditional RS.

Current state-of-the-art models in sequential recommendation (SR) comprise the usage of Recurrent Neural Networks (RNNs) [32, 56], Attention Mechanisms [26, 33] and Graph Neural Networks (GNNs) [66, 69, 72] to model the interaction sequences. In our research, we focus on GNN-approaches that construct global item graphs including all user-item interactions and learn the sequential item embedding from its neighborhood in the graph as opposed to methods that represent each user interaction sequence as a directed graph of items. An example of such global item graph construction is given in Figure 1.2. In this example the global item graph is constructed from four user interaction sequences where each transition between items increases the weight of the corresponding directed edge between those items (nodes) in the graph. Nevertheless, there are many possibilities for how to integrate the GNN framework into the task of sequential recommendation [11, 69, 71, 76].

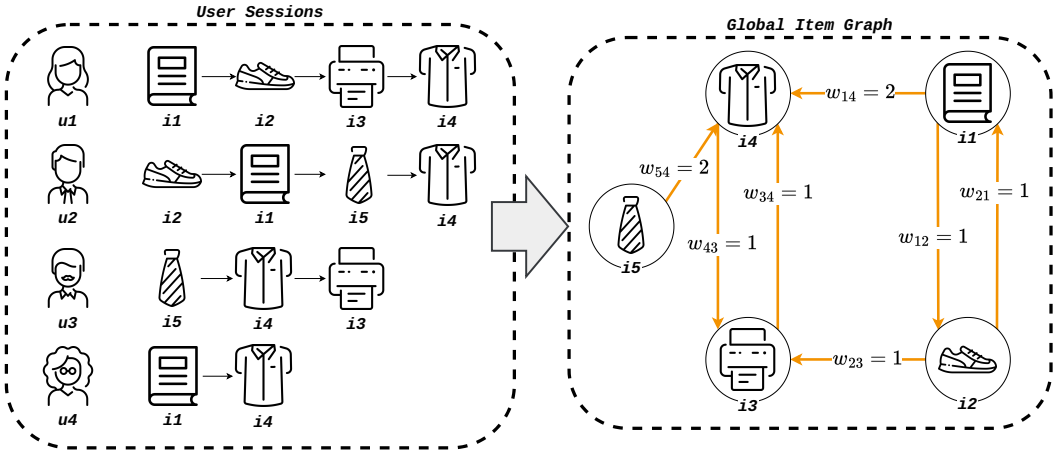


Figure 1.2.: Example of global transition graph construction from observed user behavior sequences. Edge weights correspond to the number of appearances of the item-item transitions in the user sequences. Note that the edge weights usually get normalized before used in training the GNN.

1.1. Challenges and Research Questions

We identified following important gaps in graph-based SR that we aim to tackle in this thesis:

(G1) Recent work in SR based on GNNs mostly ignore item features to improve the item representation in the model [44].

(G2) Graph-based sequential models usually only consider the order in the interaction sequence and fully neglect the dwelling time or the time difference between interaction sequences [5, 11, 31].

(G3) GNNs are prone to the over-smoothing effect, where node representations converge to the same value over multiple layers, and also introduce additional computational complexity [9, 27].

(G4) Datasets in SR potentially include noisy relations (e. g., user misclicks on an item) and can introduce misleading information into the learning process. Filtering those noisy relations on the other hand leads to an increased data sparsity, which is already severely present in the original setting of SR [17, 71].

(G5) Current works in explainability of recommender systems rely on graph-based representations [7], but struggle to provide intuitive explanations due to the lack of feature-rich datasets.

To summarize, the core goals of our research comprise extending graph representations with additional feature information as well as improving the graph construction and learning process for informative item embeddings. Our research investigates and aim to fill those described gaps in graph-based sequential recommendation. To fill the previously identified research gaps, our research seeks to address the following research questions and provide valuable contributions in this field:

RQ1: How can graphs effectively be applied to incorporate item feature information in the setting of SR? Graphs can be used in different ways in SR: To model the interaction sequences as separate graphs or to generate global item and user graphs based on the co-occurrences of item interactions, social networks, or knowledge graphs. Each node in a graph can be initially described via item feature information as opposed to simple one-hot encoding [44].

RQ2: How can graph-based methods tackle the noisy and sparse data problem? Current graph-based methods [66, 69, 71] capture the topological structure of the sequence graph and rely on multi-hop information aggregation in GNNs to exchange information along edges. Consequently, graph-based models suffer from over-smoothing (node rep-

resentations converge to the same value) if the number of layers is larger than three [9, 27]. Additionally, graph-based methods are prone to noisy item relations in the training data and introduce high complexity for large item catalogs.

RQ3: How can we effectively incorporate temporal information in the graph structure? User interaction sequences are usually not only ordered sequentially but also contain the timestamp per user-item interaction. From this information, we can infer the dwelling time or periodicity of items which potentially increases the recommendation performance. However, most of the current SRS ignore this valuable information and only rely on the order of items in a sequence [26, 58].

RQ4: How can we incorporate item features to increase recommendation performance and explainability? As described in RQ1: Feature Incorporation, each item can be described by features based on its content or meta-data. These features can support the learning process of the model by providing additional information per item. Additionally, known item features allow us to gain deeper knowledge about the insides of the model and explain its recommendation more coherently [1].

1.2. Outline

This thesis contains eight chapters, divided into two parts. Following this introductory chapter, Chapter 2 presents the related work and background on recommender systems and graph-based methods. Chapter 3 provides a summary of the papers included in this dissertation. The chapters that present technical contributions are included in Part II. This part includes Chapter 5 on unsupervised graph embeddings, Chapter 6 on contrastive graph-based shortest path search, Chapter 7 on hypergraph-based temporal modeling, and Chapter 8 on nuanced music emotion recognition. Finally, Chapter 4 in Part I presents the concluding remarks.

2. Related Work and Background

In this chapter, we provide the necessary background and review related work relevant to this doctoral thesis. We begin with an overview of core recommender systems, followed by a focus on sequential recommendation and the role of graphs in recommender systems research. We then discuss commonly used datasets and evaluation methodologies in this domain. Throughout the chapter, we identify open challenges and potential research gaps, aligning them with the overarching goal of this thesis: to advance the state of graph-based sequential recommendation.

2.1. Recommender Systems

Because people's online actions—such as what they bought or searched for—are stored electronically, it's possible to use this data to infer user preferences and predict future interests. Recommender systems significantly influence the revenue of most online platforms by delivering personalized suggestions and elevating long-tail items—items that receive infrequent interactions but can lead to high user satisfaction [34].

		Items					
		1	2	⋯	i	⋯	m
Users	1	5	3		1	2	
	2		2				4
	⋮			5			
	u	3	4		2	1	
	⋮					4	
	n			3	2		
	a	3	5	⋯	?	⋯	

Figure 2.1.: A user-item rating matrix with known ratings and a missing value $r_{a,i}$ for the active user a, which a recommender system aims to predict.

The input to a recommender system is typically a sparse matrix representing known user preferences, as illustrated in Figure 2.1. Each cell $r_{u,i}$ corresponds to the rating given by user u to item i. In practice, users rate only a tiny fraction of the available items—resulting in extremely sparse datasets, often with more than 99% of entries missing [28]. The task of a recommender system is to estimate these unknown ratings—such as $r_{a,i}$ for an active user a—and use the predicted values to recommend the most relevant items [38].

In many modern applications, however, explicit ratings are rarely available, and the data is instead binarized into implicit feedback signals (e.g., clicks, purchases, views), which are interpreted as positive interactions versus non-interactions [24].

Approaches in recommender systems can be categorized broadly into four categories [38]. Collaborative Filtering (CF) computes ratings according to past ratings of all users, whereas content-based recommender systems favor items that are similar to other items the user has rated high in the past or match with the user's attributes. Knowledge-based recommender systems use external knowledge and constraints instead of historical data to create recommendations. Hybrid approaches try to combine the strength of various types to perform more robustly in different kinds of settings [12, 63, 64]. In the following, we briefly describe the three main types of recommender system approaches—collaborative filtering, content-based, and knowledge-based—each of which employs a distinct strategy for generating recommendations.

Collaborative Filtering Models These models make use of the collaborative power of ratings provided by many users to tackle the task of recommendation. In collaborative filtering the unspecified ratings for items are imputed through the often high correlation of ratings between users and items. Take two users with similar tastes (observed by similar ratings for the same items), then it is probable that an item, which is only rated highly by one user, is also liked by the other one. There are two types of methods in collaborative filtering. Memory-based methods predict ratings of user-item combinations according to their neighborhood. The neighborhood can be either defined via users that are similar to the target user (user-based CF) or via a set S of items, containing items most similar to the target item (item-based CF) [51]. Model-based approaches apply machine learning and data mining methods to learn the parameters of the model within an optimization framework. Latent factor models like factorization machines are an example of a model-based method that covers the issue of sparse rating matrices implicitly [47]. However, collaborative filtering models generally require a large number of ratings per user to provide reliable predictions and avoid overfitting. Consequently, they are particularly affected by the cold-start problem, where insufficient user interaction data makes it difficult to generate accurate recommendations for new users [12].

Content-based Recommender Systems In content-based recommender systems the "content" of items (descriptive attributes) in combination with the ratings and buying behavior is used to generate predictions for recommendations. In the case where no access to ratings of other users is available, CF methods cannot be applied, but item descriptors can include additional information for recommendations [60]. Content-based methods use item descriptors and according ratings as training data to generate a classification or regression model specifically per each user. This model is then used to predict whether the user likes a previously unknown and unrated item or not, based on the similarity to items from the training data [39]. Despite the advantage over CF methods

in making recommendations without sufficient rating data, content-based methods lack the ability to recommend non-obvious and completely new items, since the community knowledge from similar users is not leveraged [12].

Knowledge-based Recommender Systems For items that are purchased infrequently (e.g. real estate or automobiles) and thus lack sufficient user ratings, knowledge-based recommender systems can be particularly useful. This is especially true in domains where decisions are complex and depend on specific user requirements, constraints, or expert knowledge that cannot be captured through historical interactions alone [64]. Knowledge-based methods facilitate so-called knowledge bases including rules and similarity functions based on the user requirements to perform the recommendation task. The usage of explicit user requirements results in greater control of the recommendation process in comparison to CF and content-based methods [63]. Knowledge-based methods can be distinguished into two types: Constraint-based recommender systems allow the user to specify constraints on the desired item. In case-based recommender systems the user specifies use cases which act as target or anchor points for the system [2].

Recommender systems can be applied across scenarios that differ in how they incorporate temporal dynamics and patterns of user behavior, encompassing paradigms such as general, sequential, session-based, context-aware, and social recommendation [46]. In what follows, we focus on the first three—general, sequential, and session-based—as they are most relevant to the scope of this dissertation.

General Recommendation This approach models users' long-term preferences without explicitly considering the chronological order of interactions. Each user–item interaction is treated as an independent signal of preference, aiming to capture stable and enduring interests. General recommenders are particularly effective in domains where tastes evolve slowly, such as books or durable goods, and they form the backbone of many large-scale commercial platforms [50].

Sequential Recommendation Sequential recommendation focuses on the temporal evolution of preferences, leveraging the order and recency of interactions to anticipate future choices. It assumes that recent actions are often more indicative of short-term intent, making it well suited for fast-changing domains such as news, e-commerce, and music streaming, where timeliness is critical [65]. A related setting is *session-aware recommendation*, which combines the short-term behavior within the current session with a user's longer-term history, enabling the system to balance immediate intent with enduring preferences [46].

Session-based Recommendation Session-based recommendation is a special case of sequential recommendation where the system predicts the next item based solely on the

interactions within a short, anonymous session—without access to persistent user profiles. This setting is common in contexts such as e-commerce sites without login requirements or public kiosks, where only the immediate clickstream is available. Session-based methods must capture intent from limited data, making them particularly relevant in scenarios with high privacy constraints or transient user interactions [22, 35].

2.2. Graph Neural Networks and Node Embeddings

Graph Neural Networks (GNNs) represent a powerful class of deep learning architectures specifically designed to operate on graph-structured data, where traditional neural networks fall short due to the irregular and non-Euclidean nature of graphs [52, 70]. In such data, there is no global coordinate system or uniform structure—nodes may have varying numbers of neighbors and relationships are defined by arbitrary connectivity rather than spatial proximity. Unlike conventional neural networks that process fixed-dimensional vectors or regular grids, GNNs can naturally handle variable-sized graphs with complex topological structures, making them invaluable for analyzing relational data across diverse domains.

The fundamental principle underlying GNNs is the iterative aggregation and transformation of information from neighboring nodes, allowing each node to learn representations that capture both its local features and the broader structural context within the graph [13]. The GNN message-passing framework [13] enables nodes to exchange and integrate information with their neighbors, iteratively refining their representations to capture the graph's structural and relational context. The general differentiable message passing is formulated as:

$$h_i^{(l+1)} = \sigma\left(\sum_{m \in \mathcal{M}_i} g_m\left(h_i^{(l)}, h_j^{(l)}\right)\right), \tag{2.1}$$

where $h_i^{(l)} \in \mathbb{R}^{d^{(l)}}$ represents the hidden state of node v_i at the l-th layer, with $d^{(l)}$ being the dimensionality of the layer's representation. The incoming messages, $g_m(\cdot,\cdot)$, are combined and processed through an activation function $\sigma(\cdot)$, such as ReLU. $\mathcal{M}_i$ is the set of incoming messages for node v_i, typically corresponding to the set of incoming edges. The function $g_m(\cdot,\cdot)$ is often a neural network or a simple linear transformation [27]. This transformation has proven effective in accumulating and encoding features from local, structured neighborhoods [27, 61]. Figure 2.2 illustrates the message-passing process in a GNN, where each node aggregates information from its neighbors across multiple layers, capturing both local and global structure.

GNNs have demonstrated remarkable success across numerous applications, including social network analysis [27], molecular property prediction [13], knowledge graph completion [54], recommendation systems [20], and computer vision tasks involving scene

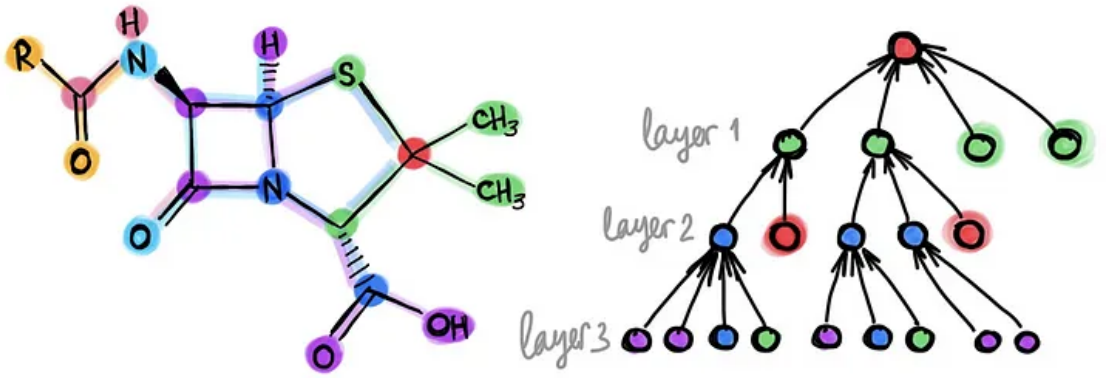


Figure 2.2.: Illustration of a molecule represented as a graph (left) and its hierarchical message-passing process in a graph neural network across layers (right). Each node aggregates information from its neighbors at increasing depths, capturing both local and global structure (taken from [6]).

graphs [70]. The power of GNNs in sequential recommendation stems from their ability to simultaneously capture multiple types of relationships within the same framework. User interaction sequences can be modeled as session graphs where items are nodes and transitions between consecutively clicked items form directed edges, enabling the capture of item-to-item transition patterns [69]. Additionally, GNNs can incorporate broader collaborative signals by constructing heterogeneous graphs that include user-item interactions, item-item similarities, and temporal dependencies, thereby combining the strengths of collaborative filtering with sequential modeling [8, 20].

The success of GNNs in capturing complex relational patterns depends critically on the quality of the underlying node representations. This has motivated the development of advanced graph embedding methods that can efficiently generate meaningful vector representations while preserving essential structural information. Graph embedding aims to generate low-dimensional vector representations of the graph's nodes which preserve topology and leverage node features. Non-deep learning methods are mainly based on random walks to explore node neighborhoods [14, 45, 57]. With Graph Convolutional Networks (GCNs) [27, 61], more sophisticated graph embedding methods than random-walk-based approaches were introduced: To scale GCNs to large graphs, the layer sampling algorithm [16] generates embeddings from a fixed node neighborhood. Current state-of-the-art methods in self-supervised/semi-supervised learning of representations rely on contrastive methods which base their loss on the difference between positive and negative samples. Deep Graph Infomax (DGI) [62] contrasts node and graph encodings by maximizing the mutual information between them. Hassani and Khasahmadi [18] propose multi-view representation learning by contrasting first-order neighbor encodings

with a general graph diffusion. Contrastive learning methods usually require a large number of negative examples and are, therefore, not scalable for large graphs. The approach by Thakoor et al. [59] learns by predicting substitute augmentations of the input and circumventing the need of contrasting with negative samples. In GraFN [30] a semi-supervised node classification framework leverages few labeled nodes to learn discriminative node representations and ensures nodes from the same class are grouped together.

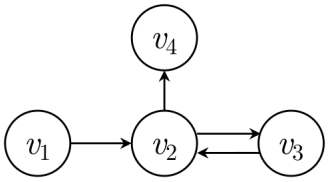
The aforementioned methods can easily incorporate external item feature information as initial node embeddings, but are rarely used in the SR domain. Additionally, none of the existing methods appear to be specifically designed for the task of auto-tagging, which aims to predict relevant labels or tags for a given item [67] and is becoming increasingly important to generate or enrich recommendation datasets (cf. gaps (G1) and (G5)).

2.3. Graph-based Sequential Recommendation Models

As described in Section 2.1, recommender systems are suited to distinct scenarios, depending on how they account for temporal dynamics and user behavior patterns. One specific scenario is sequential recommendation, which focuses on the order of user interactions over time. Sequential recommendation models aim to capture the temporal dynamics of user preferences and interactions, which can lead to more accurate and personalized recommendations.The main challenge is to model the sequential nature of user interactions, which can be represented as a sequence of items or events over time.

The initial phase of sequential recommendation focuses on discovering short-term item representations and interaction patterns. Markov decision processes are used in early works to model the interaction sequences. In FPMC [49], first-order Markov chains capture sequential patterns while matrix factorization models long-term user preferences. Also, convolutional neural networks (CNNs) have been found to be useful, where items are seen as images and short-term sequential patterns are learned via convolutional filters [58]. Xu et al. [73] combine CNNs with long-short-term memory to extract additional complex long-term dependencies. In HGN [36], a feature and instance gating mechanism is used to capture long- and short-term user interests. Other studies apply the attention mechanism to obtain and fuse different levels of interaction information [55, 77].

Self-attention and Transformer-based architectures are widely used for sequential recommendation models. SASRec [26] applies the self-attention mechanism to identify relevant interactions from the user's history. Others use custom Transformer models to provide more personalized recommendations [10, 68]. In FDSA [80], heterogeneous features of items are integrated via feature sequences, and self-attention is applied to jointly model item and feature transition patterns. S^3-Rec [82] utilizes self-supervised learning to enhance the item representations via pre-training methods.



	Outgoing edges				Incoming edges			
	1	2	3	4	1	2	3	4
1	0	1	0	0	0	0	0	0
2	0	0	1/2	1/2	1/2	0	1/2	0
3	0	1	0	0	0	1	0	0
4	0	0	0	0	0	1	0	0

Figure 2.3.: An example of a session graph and the connection matrix A_s (taken from [69]).

Recently graph-based approaches have gained popularity in sequential recommendation, leveraging the power of GNNs to capture complex relationships and dependencies within user-item interaction graphs. These methods can effectively model both short-term and long-term user preferences by constructing graphs that represent user interactions as nodes and edges. Hsu and Li [23] extract a local subgraph from a user-item pair and apply self-attention to encode long-term and short-term temporal patterns. MA-GNN [37] captures the item contextual information within a short-term period with a graph neural network and utilizes a shared memory network to model long-range dependencies. Work in [11] utilizes temporal graph representations to model continuous-time recommendation, where user and item embeddings are generated for any unseen future timestamps. Zhang et al. [81] extract augmented sequences representations from an item transition graph for a contrastive learning objective.

In session-based recommendation (SBR), a subtask of sequential recommendation, user profiles, and long-term interaction histories are no longer available. Most recent works in SBR are based on GNNs: As the first to introduce the concept of representing sessions as graphs, SR-GNN [69] models each session as a directed, unweighted graph (as shown in Figure 2.3) and applies a gating mechanism to generate session representations. This work is extended by a self-attention mechanism in GCSAN [72] to effectively capture long-range dependencies. Incorporating collaborative knowledge into GNN-based methods leads to a new line of research. GCE-GNN [66] learns item embeddings on a session level as well as on a global level and uses a soft-attention mechanism to fuse the learned item representations. Chen and Wong [9] tackle the long-range dependency (over-smoothing)

problem of session graphs by introducing a lossless encoding scheme and a shortcut graph attention layer. Xia et al. [71] introduce a dual-channel hypergraph to capture beyond-pairwise relations and apply self-supervised learning to maximize the mutual information between both session representations.

Recent research in the field of graph-based sequential recommendation has several limitations and room for improvement. Unlike earlier approaches that attempted to clean noisy data, there is little research on developing GNNs that can learn from noisy data without compromising performance (cf. gap (G4)). Additionally, there has been a recent push towards using more computationally complex GNN models that can better capture the structure and relationships within graphs. However, this increased complexity comes at the cost of greater computational resources (cf. gap (G3)). Another area of focus has been on addressing data sparsity, particularly in the context of contrastive learning (CL). Although CL has shown promise in learning representations from sparse data, there is still considerable room for improvement in this area (cf. gap (G4)).

2.4. Datasets

Sequential recommendation models are evaluated under the assumption that the underlying datasets exhibit meaningful sequential patterns. However, this assumption is often violated in practice, leading to misleading conclusions about model performance. Prior work has highlighted serious shortcomings in commonly used datasets. For example, Hidasi and Czapp [21] identify a dataset-task mismatch as a prevalent flaw, emphasizing that datasets such as MovieLens, Steam, Amazon (Beauty), and Yelp—while popular—contain weak or even artificial sequential signals due to issues like coarse timestamp granularity and presorted user histories. Similarly, others show through controlled shuffling experiments that these datasets exhibit minimal performance degradation, indicating their inadequacy for evaluating sequential recommenders [28].

Despite this, these datasets remain widely used due to legacy benchmarking and ease of access [4]. Our work explicitly avoids such flawed datasets and instead uses datasets with empirically verified sequential structure. We distinguish between two main types of datasets used:

Session-based Recommendation (SBR) These datasets capture short-term interactions within bounded sessions. Examples include Diginetica[1], Tmall[2], RetailRocket[3], Last.fm[4],

[1] https://cikm2016.cs.iupui.edu/cikm-cup/
[2] https://tianchi.aliyun.com/dataset/dataDetail?dataId=42
[3] https://www.kaggle.com/retailrocket/ecommerce-dataset
[4] http://ocelma.net/MusicRecommendationDataset/lastfm-1K.html

Music4All (Onion)[5], and Gowalla[6]. Session-based datasets often contain high-frequency temporal signals and are thus well-suited for next-click prediction tasks. Notably, Diginetica and RetailRocket, show weaker sequential signals than expected, however, their session granularity and event specificity still make them preferable for SBR evaluation [28].

Sequential Recommendation (SR) These datasets contain longer, user-centric sequences with richer temporal evolution, better suited for tasks involving long-term preference modeling. We include Ta-Feng[7], MegaMarket (SMM)[8], Delivery Hero (DHRD)[9] [3], and NowPlaying [79]. These datasets exhibit strong sequential dependencies as confirmed through both rule-mining and model-based degradation metrics [28]. For example, MegaMarket and NowPlaying demonstrate substantial drops in performance under sequence shuffling and yield low Jaccard similarity between original and perturbed top-K recommendation lists, validating their utility for sequence modeling.

Table 2.1.: Statistics of the datasets used in this thesis after preprocessing.

Dataset	**#Users/Sessions**	**#Items**	**Avg. Seq. Length**
Session-based Recommendation (SBR)			
RetailRocket	36,968	20,228	5.43
Diginetica	205,698	44,527	4.85
Tmall	377,166	40,728	6.69
Gowalla	830,893	29.510	3.85
Music4All (Onion)	601,858	80,471	7.70
Last.fm	3,510,163	38,615	11.78
Sequential Recommendation (SR)			
NowPlaying	11,310	15,905	86.39
MegaMarket (SMM)	12,098	22,167	71.97
Ta-Feng	26,162	15,642	29.99
Delivery Hero (DHRD)	42,774	20,883	12.30

In contrast to prior work using ill-suited datasets, our evaluation strategy aligns datasets with appropriate recommendation tasks, minimizing dataset-task mismatch and improving the reliability of our findings. By selecting datasets with demonstrable sequential structure, we aim to ensure that improvements in model performance genuinely reflect the model's ability to learn temporal patterns rather than artifacts of flawed benchmarks.

[5] https://zenodo.org/records/15394646
[6] https://snap.stanford.edu/data/loc-gowalla.html
[7] https://www.kaggle.com/datasets/chiranjivdas09/ta-feng-grocery-dataset
[8] https://disk.yandex.ru/d/fSEBIQYZusAAuw/datasets/data_smm
[9] https://github.com/deliveryhero/dh-reco-dataset

2.5. Evaluation and Metrics

We adopt a standard offline evaluation protocol suitable for sequential recommendation. Each user's interaction sequence is split chronologically, and the model is evaluated on its ability to predict the next item. Specifically, we use a **leave-one-out** strategy: for each user in the test set, the last interaction is withheld as ground truth, and the preceding sequence is used as input as shown in Figure 2.4. This reflects the next-item prediction task while avoiding information leakage.

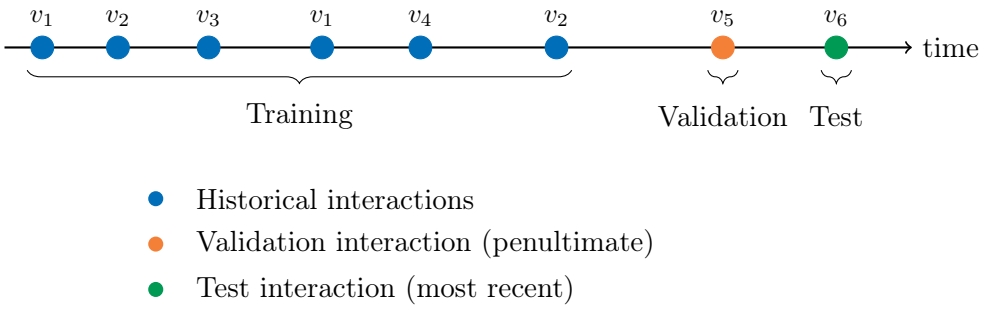


Figure 2.4.: Leave-One-Out splitting strategy.

To ensure data quality and mitigate the impact of extreme sparsity, we apply **k-core filtering**, which involves iteratively removing users and items with fewer than k interactions until all remaining users and items satisfy the threshold. This preprocessing step is widely adopted in recommender systems research to stabilize training and evaluation [22, 26, 46]. The main motivation behind k-core filtering is twofold. First, users with very few interactions provide limited signal for learning personalized preferences and can introduce noise into the model. Second, items with low interaction counts may lack sufficient co-occurrence patterns, which weakens the collaborative signal needed for both traditional and deep learning-based recommenders [78].

2.5.1. Metrics

We evaluate models using standard ranking-based metrics that capture different aspects of recommendation quality:

- **Hit Rate (HR@k)**: Measures whether the ground-truth item for a user u, denoted as item_u, appears in the top-k predicted items. Formally:

$$\text{HR@}k = \frac{1}{|U|} \sum_{u \in U} \mathbb{I}[\text{item}_u \in \hat{R}_u^k]$$

  where U is the set of users, $\hat{R}_u^k$ is the top-k recommendation list for user u, and $\mathbb{I}[\cdot]$ is the indicator function that returns 1 if the condition is true, and 0 otherwise.

- **Precision@k (P@k)**: Measures the proportion of recommended items in the top-k list that are actually relevant. For a set of users U, it is defined as:

$$\text{Precision@}k = \frac{1}{|U|} \sum_{u \in U} \frac{|\hat{R}_u^k \cap R_u|}{k}$$

  where R_u is the set of ground-truth relevant items for user u. In the common leave-one-out evaluation used for sequential recommendation, $|R_u| = 1$.

- **Normalized Discounted Cumulative Gain (NDCG@k)**: Takes into account the rank of the ground-truth item, giving higher scores when it appears near the top of the list:

$$\text{NDCG@}k = \frac{1}{|U|} \sum_{u \in U} \frac{1}{\log_2(\text{rank}_u + 1)}$$

  where rank_u is the position of the ground-truth item in $\hat{R}_u^k$. If the item does not appear in the top-k, the contribution is zero.

- **Mean Reciprocal Rank (MRR@k)**: Measures the average reciprocal rank of the first relevant item in the predicted list:

$$\text{MRR@}k = \frac{1}{|U|} \sum_{u \in U} \frac{1}{\text{rank}_u}$$

  where rank_u is the position of the ground-truth item in $\hat{R}_u^k$ if present; otherwise, the term is zero.

These metrics are computed using full-ranking evaluation (i.e., ranking all items), unless otherwise specified. We avoid negative item sampling during testing, as it introduces bias and can distort relative model performance [21, 29].

2.5.2. Limitations of Offline Evaluation

Offline evaluation offers convenience and reproducibility, but it has notable limitations that hinder its ability to reflect real-world performance accurately. A core issue is its reliance on static user preferences, while in reality, user interests evolve over time [53]. This mismatch can lead to evaluations that don't generalize well to future behavior. Another major drawback is the lack of a feedback loop. Offline metrics are computed on historical data that does not account for how recommendations influence user behavior—leading to exposure bias and a disconnect between offline and online performance [25]. Offline metrics like NDCG or Hit Rate also fail to capture broader goals such as diversity or long-term user satisfaction [78]. While offline evaluation remains a necessary benchmark, it should be complemented with online methods such as A/B testing or counterfactual estimators

to assess real-world impact. However, A/B testing is often costly, time-consuming, or impractical, due to restricted access to production systems or the need for large user bases to achieve statistical significance [21]. Overall, while offline evaluation does not fully reflect real-world deployment, it remains a valuable and necessary tool for benchmarking sequential recommender systems under controlled conditions.

3. Overview of Contributions

Throughout my doctoral studies, I have contributed to eleven research papers as an author or co-author, with four of these publications forming integral components of this dissertation. To provide readers with a comprehensive understanding of my research trajectory and scholarly contributions, I present detailed summaries of each paper below, explicitly outlining my individual role in each work and demonstrating how they address the core research questions that guide this dissertation.

For each publication, I include relevant contextual information such as conference acceptance rates and CORE rankings[1] where available, which serve as indicators of the competitive nature and quality standards of the venues where this work was published.

To maintain transparency regarding my scholarly contributions, I have systematically categorized my involvement in each paper according to three distinct dimensions: (1) the initial conception and development of the research idea, including problem identification and methodological approach; (2) the practical execution of the research work, encompassing implementation, experimentation, and data analysis; and (3) the composition and refinement of the written manuscript, including literature review, results presentation, and discussion of findings.

3.1. Unsupervised Graph Embeddings for Session-based Recommendation with Item Features

[C1] A. Peintner, M. Moscati, E. Parada-Cabaleiro, M. Schedl, and E. Zangerle. Unsupervised graph embeddings for session-based recommendation with item features. In *CARS: Workshop on Context-Aware Recommender Systems (RecSys '22)*, 2022

`RQ1: Feature Incorporation` `RQ4: Features & Explainability`

Abstract In session-based recommender systems, predictions are based on the user's preceding behavior in the session. State-of-the-art sequential recommendation algorithms either use graph neural networks to model sessions in a graph or leverage the similarity of sessions by exploiting item features. In this paper, we combine these two approaches and propose a novel method, ***G**raph **C**onvolutional **N**etwork **Ext**ension (GCNext)*, which incorporates item features directly into the graph representation via graph convolutional

[1] http://portal.core.edu.au/conf-ranks/

networks. *GCNext* creates a feature-rich item co-occurrence graph and learns the corresponding item embeddings in an unsupervised manner. We show on three datasets that integrating GCNext into sequential recommendation algorithms significantly boosts the performance of nearest–neighbor methods as well as neural network models. Our flexible extension is easy to incorporate in state-of-the-art methods and increases the *MRR@20* by up to 12.79%.

Contribution (60%, 70%, 80%) This paper was my first publication and the first one to introduce the idea of using graph convolutional networks to incorporate item features into session-based recommendation. I was primarily responsible for developing the core algorithmic framework and conducting the comprehensive experimental evaluation across multiple datasets. My involvement in the writing process was substantial, particularly in crafting the technical methodology sections and results analysis, while collaborating closely with co-authors on the literature review and discussion sections.

3.2. Efficient Session-based Recommendation with Contrastive Graph-based Shortest Path Search

[C2] A. Peintner, A. R. Mohammadi, and E. Zangerle. SPARE: shortest path global item relations for efficient session-based recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, pages 58–69. ACM, 2023. DOI: 10.1145/3604915.3608768

CORE Rank: A
Acceptance Rate: 17%

[C3] A. Peintner, A. R. Mohammadi, and E. Zangerle. Efficient session-based recommendation with contrastive graph-based shortest path search. *ACM Transactions on Recommender Systems*, 3(4), Apr. 2025. DOI: 10.1145/3701764

RQ2: Noisy & Sparse Data

Note The second paper (C3) is an extended version of the first (C2). In the remainder of this work, we refer to and use the extended version, as it provides more comprehensive experiments and deeper insights.

Abstract Session-based recommendation aims to predict the next item based on a set of anonymous sessions. Capturing user intent from a short interaction sequence imposes a variety of challenges since no user profiles are available and interaction data is naturally sparse. Recent approaches relying on graph neural networks (GNNs) for session-based

recommendation use global item relations to explore collaborative information from different sessions. These methods capture the topological structure of the graph and rely on multi-hop information aggregation in GNNs to exchange information along edges. Consequently, graph-based models suffer from noisy item relations in the training data and introduce high complexity for large item catalogs. We propose to explicitly model the multi-hop information aggregation mechanism over multiple layers via shortest-path edges based on knowledge from the sequential recommendation domain. Our approach does not require multiple layers to exchange information and ignores unreliable item-item relations. Furthermore, to address inherent data sparsity, we are the first to apply supervised contrastive learning by mining data-driven positive and hard negative item samples from the training data. Extensive experiments on four different datasets show that the proposed approach outperforms almost all of the state-of-the-art methods.

Contribution (90%, 80%, 60%) *This contribution is reflected in both the RecSys 2023 conference paper and its extended version published in ACM TORS 2025.*

In this work, which addresses the challenges of noisy and sparse data in session-based recommendation, I led the development of the core methodology by proposing the combination of shortest-path edge modeling with supervised contrastive learning. I was primarily responsible for implementing the proposed method and carrying out an extensive experimental evaluation across four benchmark datasets. My writing contributions focused on the methodology and experimental sections, while I collaborated closely with co-authors on the theoretical background and related work. The core ideas were first introduced in our RecSys 2023 paper and subsequently extended and refined in the TORS 2025 journal version.

3.3. Hypergraph-based Temporal Modelling of Repeated Intent for Sequential Recommendation

[C4] A. Peintner, A. R. Mohammadi, M. Müller, and E. Zangerle. Hypergraph-based temporal modelling of repeated intent for sequential recommendation. In *Proceedings of the ACM on Web Conference 2025, WWW 2025*, pages 3809–3818. ACM, 2025. DOI: 10.1145/3696410.3714896

CORE Rank: A*
Acceptance Rate: 19.3%

RQ3: Temporal Information

Abstract In sequential recommendation scenarios, user intent is a key driver of consumption behavior. However, consumption intents are usually latent and hence, difficult to

leverage for recommender systems. Additionally, intents can be of repeated nature (e. g., yearly shopping for christmas gifts or buying a new phone), which has not been exploited by previous approaches. To navigate these impediments we propose the *HyperHawkes* model which models user sessions via hypergraphs and extracts user intents via soft clustering. We use Hawkes Processes to model the temporal dynamics of intents, namely repeated consumption patterns and long-term interests of users. For short-term interest adaption, which is more fine-grained than intent-level modeling, we use a multi-level attention mixture network and fuse long-term and short-term signals. We use the generalized expectation-maximization (EM) framework for training the model by alternating between intent representation learning and optimizing parameters of the long- and short-term modules. Extensive experiments on four real-world datasets from different domains show that HyperHawkes significantly outperforms existing state-of-the-art methods.

Contribution (80%, 90%, 60%) This paper on temporal information modeling through the HyperHawkes model represents one of my most significant contributions to the field. I conceived the core research idea of combining hypergraphs with Hawkes processes for modeling temporal dynamics in user behavior. I led the implementation of the complex model architecture, including the generalized EM framework for training, and conducted comprehensive experiments across four real-world datasets. I was the primary author of the paper, taking responsibility for most sections while incorporating valuable feedback and contributions from co-authors.

3.4. Nuanced Music Emotion Recognition via a Semi-Supervised Multi-Relational Graph Neural Network

[C5] A. Peintner, M. Moscati, Y. Kinoshita, R. Vogl, P. Knees, M. Schedl, H. Strauss, M. Zentner, and E. Zangerle. Nuanced music emotion recognition via a semi-supervised multi-relational graph neural network. *Transactions of the International Society for Music Information Retrieval*, 8(1):140–153, 2025. DOI: 10.5334/tismir.235

RQ1: Feature Incorporation | RQ4: Features & Explainability

Abstract Music Emotion Recognition (MER) seeks to understand the complex emotional landscapes elicited by music, acknowledging music's profound social and psychological roles beyond traditional tasks such as genre classification or content similarity. MER relies heavily on high-quality emotional annotations, which provide the foundation for training models to recognize emotions. However, collecting these annotations is both complex and costly, leading to limited availability of large-scale datasets for MER. Recent works in MER for automatically extracting emotion aim to learn track representations in a supervised manner. However, these approaches mainly utilize simpler emotion

models due to limited datasets or the lack of necessity of sophisticated emotion models and ignore hidden inter-track relations, which are beneficial for a semi-supervised learning setting. This paper proposes a novel approach to MER by constructing a multi-relational graph that encapsulates different facets of music. We leverage Graph Neural Networks (GNNs) to model intricate inter-track relationships and capture structurally induced representations from user data, such as listening histories, genres and tags. Our model, the Semi-supervised Multi-relational Graph Neural Network for Emotion Recognition (SRGNN-Emo), innovates by combining graph-based modeling with semi-supervised learning, using rich user data to extract nuanced emotional profiles from music tracks. Through extensive experimentation, SRGNN-Emo achieves significant improvements in R^2 and RMSE metrics for predicting the intensity of nine continuous emotions (GEMS), demonstrating its superior capability in capturing and predicting complex emotional expressions in music.

Contribution (80%, 70%, 40%) In this music emotion recognition paper, I led the development of SRGNN-Emo, including its conceptual design and full implementation. I carried out the experiments for emotion prediction tasks and designed the evaluation pipeline. I authored the technical methodology and experimental design sections. The co-authors contributed domain knowledge in music information retrieval and emotional modeling, which informed the interpretation of results and theoretical framing.

3.5. Not-included Contributions

The following enumeration presents additional research articles that were developed during the course of this doctoral work, which have either been published or are presently under peer review. While these contributions represent valuable scholarly output from my doctoral research period, they address topics that fall outside the primary scope and thematic focus of this dissertation. Consequently, these works are not incorporated into the main body of this thesis, though they demonstrate the breadth of research activities undertaken during my doctoral studies.

[C6] L. Benning, A. Peintner, G. Finkenzeller, and L. Peintner. Automated spheroid generation, drug application and efficacy screening using a deep learning classification: a feasibility study. *Scientific Reports*, 10(1):1–11, 2020

[C7] L. Benning, A. Peintner, and L. Peintner. Advances in and the applicability of machine learning-based screening and early detection approaches for cancer: a primer. *Cancers*, 14(3):623, 2022

[C8] A. Peintner. Sequential recommendation models: A graph-based perspective. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, pages 1295–1299. ACM, 2023

[C9] M. Moscati, H. Strauß, P.-O. Jacobsen, A. Peintner, E. Zangerle, M. Zentner, and M. Schedl. Emotion-based music recommendation from quality annotations and large-scale user-generated tags. In *Proceedings of the 32nd ACM Conference on User Modeling, Adaptation and Personalization*, pages 159–164, 2024

[C10] A. R. Mohammadi, A. Peintner, M. Müller, and E. Zangerle. Are we explaining the same recommenders? Incorporating recommender performance for evaluating explainers. In *Proceedings of the 18th ACM Conference on Recommender Systems, RecSys 2024*, pages 1113–1118. ACM, 2024

[C11] S. Ewerz and A. Peintner. Unternehmensleitung in der Aktiengesellschaft und künstliche Intelligenz. *Wirtschaftsrechtliche Blätter: WBL; Zeitschrift für österreichisches und europäisches Wirtschaftsrecht*:697–710, 2024

4. Conclusion

This dissertation embarked on an investigation into the challenges and opportunities within graph-based sequential recommendation. The primary objective was to extend the capabilities of sequential recommender systems by developing novel graph-based methodologies that address critical gaps in the literature. Our research was guided by four central questions focused on incorporating item features (RQ1: Feature Incorporation), tackling data sparsity and noise (RQ2: Noisy & Sparse Data), modeling complex temporal dynamics (RQ3: Temporal Information), and enhancing recommendation explainability (RQ4: Features & Explainability). Through a series of publications, we have introduced new models and frameworks that provide substantial answers to these questions, pushing the boundaries of what is possible in sequential recommendation. This concluding chapter summarizes the key contributions of this thesis, discusses the inherent limitations of the proposed methods, and outlines promising directions for future research.

4.1. Summary of Contributions

The research presented in this dissertation advanced the state-of-the-art through several key methodological contributions, each aligned with our guiding research questions.

In response to RQ1: Feature Incorporation and RQ4: Features & Explainability, our work demonstrated that item features can be powerfully integrated into sequential models by leveraging the structural properties of graphs. We established that learning item embeddings from a global co-occurrence graph in an unsupervised manner, where nodes are enriched with feature information, provides a robust initialization for downstream sequential models. This approach significantly boosts performance by encoding both collaborative signals and content-based semantics. We further extended this principle to more complex, multi-relational graph structures, showing that by modeling diverse relationships (e.g., shared genres, user sessions, tags) in a semi-supervised framework, we can effectively predict nuanced, multi-dimensional attributes such as a music track's emotional profile. This underscores the versatility of graph-based feature integration for both core recommendation and related, feature-dependent tasks.

To contend with the pervasive issues of noise and data sparsity (RQ2: Noisy & Sparse Data), we proposed a novel graph construction paradigm. We showed that by pruning a global item graph based on shortest-path distances, it is possible to filter out noisy, low-support relations while explicitly creating shortcut connections that model multi-hop dependencies. This strategy not only improves the robustness of the learned representations but also yields a more efficient model architecture that avoids the complexities

of deep GNNs. To mitigate the risk of increased data sparsity from this pruning, we introduced a supervised contrastive learning objective. By mining informative positive and hard-negative samples directly from the training data, this component effectively refines item embeddings and enhances the model's ability to discriminate between closely related items.

Addressing the challenge of modeling complex temporal dynamics (RQ3: Temporal Information), this thesis introduced a framework that moves beyond item-level sequences to capture the temporal evolution of latent user intents. We demonstrated that by representing user sessions as hyperedges, we can capture higher-order item relationships indicative of a common intent. These latent intents, extracted via soft clustering, become the primary unit for temporal analysis. By applying temporal point processes, specifically Hawkes Processes [19], we modeled the self-exciting nature of these intents, thereby capturing sophisticated patterns such as repeat consumption and periodicity. This long-term, intent-driven perspective, when fused with a more traditional short-term attention mechanism, provides a more comprehensive and accurate model of user behavior over time.

4.2. Limitations and Future Directions

Despite the substantial contributions to the field, the methods proposed in this dissertation have several limitations. These challenges, however, pave the way for several exciting avenues for future research in graph-based sequential recommendation.

A significant limitation of the proposed methods is their reliance on static, offline graph representations. The initial graph construction, particularly processes like all-pairs shortest-path search or frequent itemset mining, can be a computational bottleneck at scale. More importantly, these static graphs do not reflect the dynamic nature of real-world recommender systems where new items, users, and interactions are constantly streaming in [15]. A critical next step is to adapt these frameworks for dynamic environments. This involves developing algorithms for efficient, real-time updates to the graph structure and embeddings as new data arrives. Techniques from continual learning and incremental graph processing will be instrumental in creating models that can evolve with user behavior and item catalogs without periodic, costly retraining [74].

While RQ4: Features & Explainability touched upon explainability, the primary focus of our contributions was on leveraging features to improve predictive accuracy. The inherent "black-box" nature of the GNNs used persists. The models can show that certain items are related but struggle to provide intuitive, human-understandable reasons why a specific recommendation was made [1, 7]. Building truly explainable graph-based recommender systems is a major challenge for future work. This research could focus on designing inherently interpretable GNN architectures or developing post-hoc explanation

techniques. A promising direction is to identify and present the critical subgraphs or interaction paths most influential in a recommendation, effectively translating the model's complex reasoning into a narrative that users can trust and understand [75].

In conclusion, this dissertation has demonstrated the immense potential of viewing sequential recommendation through a graph-based lens. By developing novel methods for feature integration, noise reduction, and temporal modeling, we have made significant strides in the field. The path forward is rich with challenges and opportunities, and it is our hope that the work presented here will serve as a solid foundation for the next generation of intelligent, fair, and explainable recommender systems.

Bibliography

[1] D. Afchar, A. Melchiorre, M. Schedl, R. Hennequin, E. Epure, and M. Moussallam. Explainability in music recommender systems. *AI Magazine*, 43(2):190–208, 2022.

[2] C. C. Aggarwal. *Recommender Systems - The Textbook*. Springer, 2016.

[3] Y. Assylbekov, R. Bali, L. Bovard, and C. Klaue. Delivery hero recommendation dataset: A novel dataset for benchmarking recommendation algorithms. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, pages 1042–1044. ACM, 2023.

[4] C. Bauer, E. Zangerle, and A. Said. Exploring the landscape of recommender systems evaluation: practices and perspectives. *ACM Transactions on Recommender Systems*, 2(1):1–31, 2024.

[5] V. Bogina, T. Kuflik, D. Jannach, M. Bieliková, M. Kompan, and C. Trattner. Considering temporal aspects in recommender systems: a survey. *User Model. User Adapt. Interact.*, 33(1):81–119, 2023.

[6] M. Bronstein. Do we need deep graph neural networks? Medium, July 2020. URL: https://medium.com/data-science/do-we-need-deep-graph-neural-networks-be62d3ec5c59.

[7] H. Chen, Y. Li, X. Sun, G. Xu, and H. Yin. Temporal meta-path guided explainable recommendation. In *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining*, pages 1056–1064. ACM, 2021.

[8] L. Chen, L. Wu, R. Hong, K. Zhang, and M. Wang. Revisiting graph based collaborative filtering: a linear residual graph convolutional network approach. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34 of number 01, pages 27–34, 2020.

[9] T. Chen and R. C. Wong. Handling information loss of graph neural networks for session-based recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1172–1180. ACM, 2020.

[10] G. de Souza Pereira Moreira, S. Rabhi, J. M. Lee, R. Ak, and E. Oldridge. Transformers4rec: bridging the gap between NLP and sequential / session-based recommendation. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems*, pages 143–153. ACM, 2021.

[11] Z. Fan, Z. Liu, J. Zhang, Y. Xiong, L. Zheng, and P. S. Yu. Continuous-time sequential recommendation with temporal graph collaborative transformer. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*, pages 433–442. ACM, 2021.

[12] C. Ganhör, M. Moscati, A. Hausberger, S. Nawaz, and M. Schedl. A multimodal single-branch embedding network for recommendation in cold-start and missing modality scenarios. In *Proceedings of the 18th ACM Conference on Recommender Systems, RecSys 2024*, pages 380–390. ACM, 2024.

[13] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[14] A. Grover and J. Leskovec. Node2vec: scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 2016*, pages 855–864. ACM, 2016.

[15] L. Guo, H. Yin, Q. Wang, T. Chen, A. Zhou, and N. Q. V. Hung. Streaming session-based recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019*, pages 1569–1577. ACM, 2019.

[16] W. L. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 1024–1034, 2017.

[17] Q. Han, C. Zhang, R. Chen, R. Lai, H. Song, and L. Li. Multi-faceted global item relation learning for session-based recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1705–1715. ACM, 2022.

[18] K. Hassani and A. H. K. Ahmadi. Contrastive multi-view representation learning on graphs. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, volume 119 of *Proceedings of Machine Learning Research*, pages 4116–4126. PMLR, 2020.

[19] A. G. Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

[20] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang. Lightgcn: simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, pages 639–648. ACM, 2020.

[21] B. Hidasi and Á. T. Czapp. Widespread flaws in offline evaluation of recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, pages 848–855. ACM, 2023.

[22] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, Conference Track Proceedings*, 2016.

[23] C. Hsu and C. Li. Retagnn: relational temporal attentive graph neural networks for holistic sequential recommendation. In *WWW '21: The Web Conference 2021*, pages 2968–2979. ACM, 2021.

[24] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008)*, pages 263–272. IEEE Computer Society, 2008.

[25] T. Joachims, A. Swaminathan, and T. Schnabel. Unbiased learning-to-rank with biased feedback. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 781–789, 2017.

[26] W. Kang and J. J. McAuley. Self-attentive sequential recommendation. In *IEEE International Conference on Data Mining, ICDM 2018*, pages 197–206. IEEE Computer Society, 2018.

[27] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.

[28] A. Klenitskiy, A. Volodkevich, A. Pembek, and A. Vasilev. Does it look sequential? an analysis of datasets for evaluation of sequential recommendations. In *Proceedings of the 18th ACM Conference on Recommender Systems*, RecSys '24, pages 1067–1072. ACM, 2024.

[29] W. Krichene and S. Rendle. On sampled metrics for item recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1748–1757. ACM, 2020.

[30] J. Lee, Y. Oh, Y. In, N. Lee, D. Hyun, and C. Park. Grafn: semi-supervised node classification on graph with few labels via non-parametric distribution assignment. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2243–2248. ACM, 2022.

[31] J. Li, Y. Wang, and J. J. McAuley. Time interval aware self-attention for sequential recommendation. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining*, pages 322–330. ACM, 2020.

[32] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017*, pages 1419–1428. ACM, 2017.

[33] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018*, pages 1831–1839. ACM, 2018.

[34] L. Lü, M. Medo, C. H. Yeung, Y.-C. Zhang, Z.-K. Zhang, and T. Zhou. Recommender systems. *Physics reports*, 519(1):1–49, 2012.

[35] M. Ludewig, I. Kamehkhosh, N. Landia, and D. Jannach. Effective nearest-neighbor music recommendations. In *Proceedings of the ACM Recommender Systems Challenge, RecSys Challenge 2018*, 3:1–3:6. ACM, 2018.

[36] C. Ma, P. Kang, and X. Liu. Hierarchical gating networks for sequential recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019*, pages 825–833. ACM, 2019.

[37] C. Ma, L. Ma, Y. Zhang, J. Sun, X. Liu, and M. Coates. Memory augmented graph neural networks for sequential recommendation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pages 5045–5052. AAAI Press, 2020.

[38] P. Melville and V. Sindhwani. Recommender systems. *Encyclopedia of machine learning*, 1:829–838, 2010.

[39] M. Moscati, E. Parada-Cabaleiro, Y. Deldjoo, E. Zangerle, and M. Schedl. Music4all-onion - A large-scale multi-faceted content-centric music recommendation dataset. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4339–4343. ACM, 2022.

[40] A. Peintner, A. R. Mohammadi, M. Müller, and E. Zangerle. Hypergraph-based temporal modelling of repeated intent for sequential recommendation. In *Proceedings of the ACM on Web Conference 2025, WWW 2025*, pages 3809–3818. ACM, 2025. DOI: 10.1145/3696410.3714896.

[41] A. Peintner, A. R. Mohammadi, and E. Zangerle. Efficient session-based recommendation with contrastive graph-based shortest path search. *ACM Transactions on Recommender Systems*, 3(4), Apr. 2025. DOI: 10.1145/3701764.

[42] A. Peintner, A. R. Mohammadi, and E. Zangerle. SPARE: shortest path global item relations for efficient session-based recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, pages 58–69. ACM, 2023. DOI: 10.1145/3604915.3608768.

[43] A. Peintner, M. Moscati, Y. Kinoshita, R. Vogl, P. Knees, M. Schedl, H. Strauss, M. Zentner, and E. Zangerle. Nuanced music emotion recognition via a semi-supervised multi-relational graph neural network. *Transactions of the International Society for Music Information Retrieval*, 8(1):140–153, 2025. DOI: 10.5334/tismir.235.

[44] A. Peintner, M. Moscati, E. Parada-Cabaleiro, M. Schedl, and E. Zangerle. Unsupervised graph embeddings for session-based recommendation with item features. In *CARS: Workshop on Context-Aware Recommender Systems (RecSys '22)*, 2022.

[45] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: online learning of social representations. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14*, pages 701–710. ACM, 2014.

[46] M. Quadrana, P. Cremonesi, and D. Jannach. Sequence-aware recommender systems. *ACM Computing Surveys (CSUR)*, 51(4):1–36, 2018.

[47] S. Rendle. Factorization machines. In *2010 IEEE International conference on data mining*, pages 995–1000. IEEE, 2010.

[48] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461. AUAI Press, 2009.

[49] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010*, pages 811–820. ACM, 2010.

[50] F. Ricci, L. Rokach, and B. Shapira. Introduction to recommender systems handbook. In *Recommender Systems Handbook*, pages 1–35. Springer, 2011.

[51] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10*, pages 285–295. ACM, 2001.

[52] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[53] M. Schedl, H. Zamani, C.-W. Chen, Y. Deldjoo, and M. Elahi. Current challenges and visions in music recommender systems research. *International Journal of Multimedia Information Retrieval*, 7:95–116, 2018.

[54] M. S. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer, 2018.

[55] Q. Tan, J. Zhang, N. Liu, X. Huang, H. Yang, J. Zhou, and X. Hu. Dynamic memory based attention network for sequential recommendation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, pages 4384–4392. AAAI Press, 2021.

[56] Y. K. Tan, X. Xu, and Y. Liu. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS@RecSys 2016*, pages 17–22. ACM, 2016.

[57] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. LINE: large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015*, pages 1067–1077. ACM, 2015.

[58] J. Tang and K. Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018*, pages 565–573. ACM, 2018.

[59] S. Thakoor, C. Tallec, M. G. Azar, R. Munos, P. Veličković, and M. Valko. Bootstrapped representation learning on graphs. In *ICLR 2021 Workshop on Geometrical and Topological Representation Learning*, 2021.

[60] T. X. Tuan and T. M. Phuong. 3d convolutional networks for session-based recommendation with content features. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017*, pages 138–146. ACM, 2017.

[61] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[62] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm. Deep graph infomax. In *7th International Conference on Learning Representations, ICLR 2019*. OpenReview.net, 2019.

[63] B. Wang and W. Cai. Knowledge-enhanced graph neural networks for sequential recommendation. *Inf.*, 11(8):388, 2020.

[64] C. Wang, M. Zhang, W. Ma, Y. Liu, and S. Ma. Make it a chorus: knowledge- and time-aware item modeling for sequential recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, pages 109–118. ACM, 2020.

[65] S. Wang, L. Hu, Y. Wang, L. Cao, Q. Z. Sheng, and M. Orgun. Sequential recommender systems: challenges, progress and prospects. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 6332–6338, 2019.

[66] Z. Wang, W. Wei, G. Cong, X. Li, X. Mao, and M. Qiu. Global context enhanced graph neural networks for session-based recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, pages 169–178. ACM, 2020.

[67] M. Won, A. Ferraro, D. Bogdanov, and X. Serra. Evaluation of cnn-based automatic music tagging models. *CoRR*, abs/2006.00751, 2020.

[68] L. Wu, S. Li, C. Hsieh, and J. Sharpnack. SSE-PT: sequential recommendation via personalized transformer. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems*, pages 328–337. ACM, 2020.

[69] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan. Session-based recommendation with graph neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, volume 33 of number 01, pages 346–353. AAAI Press, July 2019.

[70] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[71] X. Xia, H. Yin, J. Yu, Y. Shao, and L. Cui. Self-supervised graph co-training for session-based recommendation. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*, pages 2180–2190. ACM, 2021.

[72] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, and X. Zhou. Graph contextualized self-attention network for session-based recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 3940–3946. ijcai.org, 2019.

[73] C. Xu, P. Zhao, Y. Liu, J. Xu, V. S. Sheng, Z. Cui, X. Zhou, and H. Xiong. Recurrent convolutional neural network for sequential recommendation. In *The World Wide Web Conference, WWW 2019*, pages 3398–3404. ACM, 2019.

[74] Y. Xu, Y. Zhang, W. Guo, H. Guo, R. Tang, and M. Coates. Graphsail: graph structure aware incremental learning for recommender systems. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management*, pages 2861–2868. ACM, 2020.

[75] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec. Gnnexplainer: generating explanations for graph neural networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 9240–9251.

[76] J. Yu, H. Yin, X. Xia, T. Chen, L. Cui, and Q. V. H. Nguyen. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In SIGIR '22, pages 1294–1303. ACM, 2022.

[77] L. Yu, C. Zhang, S. Liang, and X. Zhang. Multi-order attentive ranking model for sequential recommendation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 5709–5716. AAAI Press, 2019.

[78] E. Zangerle and C. Bauer. Evaluating Recommender Systems: Survey and Framework. *ACM Computing Surveys*, 55(8):170:1–170:38, 2022.

[79] E. Zangerle, M. Pichl, W. Gassler, and G. Specht. #nowplaying music dataset: extracting listening behavior from twitter. In *Proceedings of the First International Workshop on Internet-Scale Multimedia Management, WISMM 2014*, pages 21–26. ACM, 2014.

[80] T. Zhang, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, D. Wang, G. Liu, and X. Zhou. Feature-level deeper self-attention network for sequential recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 4320–4326. ijcai.org, 2019.

[81] Y. Zhang, Y. Liu, Y. Xu, H. Xiong, C. Lei, W. He, L. Cui, and C. Miao. Enhancing sequential recommendation with graph contrastive learning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022*, pages 2398–2405. ijcai.org, 2022.

[82] K. Zhou, H. Wang, W. X. Zhao, Y. Zhu, S. Wang, F. Zhang, Z. Wang, and J. Wen. S3-rec: self-supervised learning for sequential recommendation with mutual information maximization. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management*, pages 1893–1902. ACM, 2020.

Part II.

Selected Papers

5. Unsupervised Graph Embeddings for Session-based Recommendation with Item Features

Publication

A. Peintner, M. Moscati, E. Parada-Cabaleiro, M. Schedl, and E. Zangerle. Unsupervised graph embeddings for session-based recommendation with item features. In *CARS: Workshop on Context-Aware Recommender Systems (RecSys '22)*, 2022

Abstract

In session-based recommender systems, predictions are based on the user's preceding behavior in the session. State-of-the-art sequential recommendation algorithms either use graph neural networks to model sessions in a graph or leverage the similarity of sessions by exploiting item features. In this paper, we combine these two approaches and propose a novel method, ***G**raph **C**onvolutional **N**etwork **Ext**ension (GCNext)*, which incorporates item features directly into the graph representation via graph convolutional networks. *GCNext* creates a feature-rich item co-occurrence graph and learns the corresponding item embeddings in an unsupervised manner. We show on three datasets that integrating GCNext into sequential recommendation algorithms significantly boosts the performance of nearest–neighbor methods as well as neural network models. Our flexible extension is easy to incorporate in state-of-the-art methods and increases the *MRR@20* by up to 12.79%.

5.1. Introduction

Recommender systems (RecSys) traditionally leverage the users' rich interaction data with the system. However, in some cases, such data are not available. Session-based recommender systems, in contrast, aim to predict the next item the user will interact with (e. g., click on, purchase, or listen to) only based on the preceding interactions in the current session. The task of session-based recommendation can be defined as follows. Consider the set X of all items in the catalog, $x \in X$ being an individual item, and $m = |X|$ being the number of items in the catalog. Given an interaction session $[x_1, x_2, ..., x_n]$ (ordered by timestamp), the goal is to predict a ranked list $[y_1, y_2, ..., y_m]$ of items with corresponding relevance scores to continue the session. The top-k values of the ranked list are chosen as recommendation candidates. As opposed to session-aware or sequential recommendations, the inputs to session-based RecSys are only items of the current session and their features; users are anonymous and no inter-session data is available.

Current approaches for session-based recommendation leverage Recurrent Neural Networks (RNNs) [14, 31, 34], attention networks [20, 21], Graph Neural Networks (GNNs) [27, 41, 42], or transformer architectures [4, 17, 33]. Also, classical nearest–neighbor methods have been used [6, 16, 24, 25]. Most current methods focus on the sequential nature of sessions; RNNs and nearest–neighbor methods have dominated research in the past few years. Extensions to these models use additional item features to enrich the item representations. Item features capture contextual information (e. g., item category) which is relevant to the task of session-based recommendation, which itself can be considered a special case of context-aware recommender systems [30]. However, recently, GNN models have been shown to outperform RNN- and nearest-neighbor–based methods [9, 25, 41]. Yet, to the best of our knowledge, no approach combines auxiliary item features and GNNs to learn informative embeddings for sequential models. In this paper, we therefore propose ***G**raph **C**onvolutional **N**etwork **Ext**ension (GCNext)*, which extracts node embeddings from a feature-rich item co-occurrence graph via unsupervised learning with Graph Convolutional Networks (GCNs). We then use these pre-trained item embeddings as auxiliary features describing items and their structural dependence. One major advantage and novelty of GCNext is that it can flexibly be plugged into any current sequential recommendation method. Particularly, we (1) use the computed item embeddings to initialize sequential neural network models, and (2) extend (non-neural) nearest–neighborhood methods with pre-trained item graph embeddings to refine the search of candidate sessions for recommendation.

Our main contributions are as follows: (1) We present GCNext, a novel method for session-based recommendation based on a item co-occurence graph for sessions. GCNext combines the topological representation power of GCNs with the session representation generated by neural network sequential models without modifying their architecture; (2) GCNext can easily extend nearest-neighbor methods as well as neural network models in a plug-in fashion to further enhance the performance of these models; (3) We perform

a large-scale evaluation of graph-based item embeddings and their impact on a diverse set of sequential models. We find that adding GCNext is not only able to boost the performance of current methods, but also shows significant performance improvements over current state-of-the-art sequential models.

5.2. Related Work

In the following, we briefly present related works in the field of graph and node embedding. We subsequently discuss approaches for sequential and session-based recommendation, which incorporate side information or GNNs.

5.2.1. Graph and Node Embeddings

Graph embedding aims to generate low-dimensional vector representations of the graph's nodes which preserve topology and leverage node features. Non-deep learning methods are mainly based on random walks to explore node neighborhoods [8, 28, 35]. With GCNs [19, 38], more sophisticated graph embedding methods were introduced: To scale GCNs to large graphs, the layer sampling algorithm [10] generates embeddings from a fixed node neighborhood. Current state-of-the-art methods in unsupervised learning of representations rely on contrastive methods which base their loss on the difference between positive and negative samples. Deep Graph Infomax (DGI) [39] contrasts node and graph encodings by maximizing the mutual information between them. Hassani and Khasahmadi [11] propose multi-view representation learning by contrasting first-order neighbor encodings with a general graph diffusion. Contrastive learning methods usually require a large number of negative examples and are, therefore, not scalable for large graphs. The approach by Thakoor et al. [37] learns by predicting substitute augmentations of the input and circumventing the need of contrasting with negative samples.

5.2.2. Sequential Recommendation

Non-neural sequential recommendation approaches focus on the similarity of sessions to extract potential next items. Several works extend the session-based nearest-neighbors method with additional factors such as positions, recency, and popularity [6, 16, 24, 25]. Other works [17, 20, 21, 34] model item-to-item transitions using neural networks, possibly incorporating item features [4, 15, 43].

Recent works exploit the graph-based representation of sessions for improved recommendations. Current state-of-the-art use GNNs—in combination with attention or self-attention modules—to capture complex transitions and rich local dependencies [41, 42].

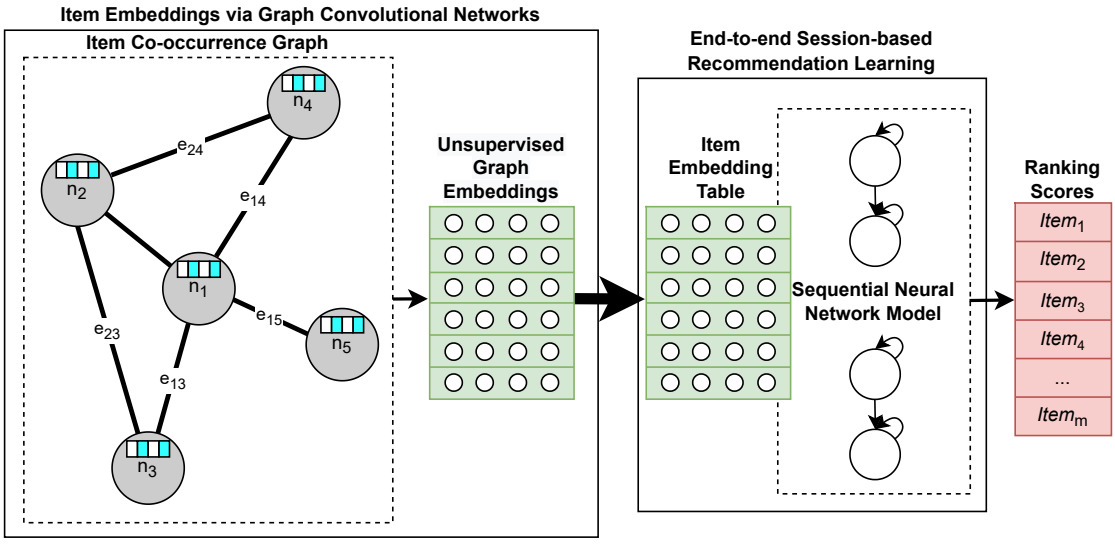
Further approaches enrich the graph topology with knowledge base entities [1, 40]. Gwadabe and Liu [9] use an item co-occurrence graph to generate session co-occurrence representations which are combined with the local and global preferences of users.

In contrast to models that integrate item features by extending the network with additional paths, GCNext extracts item embeddings from the item co-occurrence graph, in which content-based features are attached to each node. GCNext can be added to different sequential models without modifying their architecture, essentially using it in a plug-in fashion. Compared to already existing graph-based pretraining schemes [22, 26] for general recommendation, our approach specifically tackles the task of sequential and session-based recommendations.

5.3. Graph Convolutional Network Extension (GCNext)

In this section, we present the proposed GCNext approach. An overview of GCNext applied to sequential neural network models is shown in Figure 5.1. The first component represents the item co-occurrence graph from which we extract corresponding node embeddings by applying an unsupervised learning method. We subsequently use these embeddings to initialize the item embedding table of the underlying end-to-end sequential model, which learns session-based recommendations. Furthermore, we show how GCNext can also be employed for nearest–neighbor methods.

Figure 5.1.: Overview of the graph-based generation of item embeddings and its application in sequential neural network models.



5.3.1. Unsupervised Graph Embeddings

In contrast to other graph-based models [41, 42], we generate the item-item graph by extracting item co-occurrences in sessions. Each item is modeled as a node with the item's features as descriptors. For two nodes n_1 and n_2, an edge between the nodes represents the co-occurrence of two items i_1 and i_2 in a session. Edges are undirected, which prevents sequential information from being modeled in the graph. Each edge is weighted by $\mathbf{e}_{ij}$ which denotes the normalized number of co-occurrences (in all sessions) of the two items it connects. The learning of item-item transitions is solely the task of the underlying sequential model and the item-item graph embedding only incorporates the similarity to other items. We strictly rely on inductive representations for new items to obviate any data leakage into the embeddings.

The item catalog can contain millions of unique items which strongly impacts the scalability of the generated item-item graph. Therefore, we apply Bootstrapped Graph Latents (BGRL) [37] that can be scaled up to graphs with hundreds of millions of nodes and reduce the memory requirement substantially by enriching the original graph with simple augmentations to produce two different, but semantically similar views. Two encoders then generate online and target embeddings. The online embedding is used as input to a predictor which forms a prediction for the target embedding. The cosine similarity of the predictor output and the generated target embedding by the encoder is the final objective. Our graph embedding in GCNext is based on the BGRL learning method with our custom encoder architecture. We use the attentional convolution as introduced in [38] and optimized in [2] as *GATv2*. The node-wise formulation of this graph operation is defined as:

$$\mathbf{h}_i' = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \cdot \mathbf{W}\mathbf{h}_j\right), \tag{5.1}$$

where $\mathbf{h}$ denotes the node features, $\mathbf{W}$ the linear transformation's weight matrix and the average over the neighbor nodes features is weighted by the normalized attention weights α_{ij}. The computation of the normalized attention coefficients with softmax including edge weights can be seen in:

$$\alpha_{ij} = \frac{\exp\left(\mathbf{a}^\top \text{LeakyReLU}\left(\mathbf{W} \cdot [\mathbf{h}_i \,\|\, \mathbf{h}_j \,\|\, \mathbf{e}_{ij}]\right)\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\mathbf{a}^\top \text{LeakyReLU}\left(\mathbf{W} \cdot [\mathbf{h}_i \,\|\, \mathbf{h}_k \,\|\, \mathbf{e}_{ik}]\right)\right)}, \tag{5.2}$$

where $\mathbf{W}$ is a learnable weight matrix, $\mathbf{h}$ again denotes the corresponding node features, $\mathbf{e}_{ij}$ is the edge weight from node n_i to neighboring node n_j and $\|$ denotes vector concatenation.

Along the lines of other encoder architectures [37, 39], we stack multiple graph convolutional layers using skip connections and PReLU [12] as an activation function. The second layer of the encoder performs the computation:

$$\mathbf{H_2} = \sigma(\text{GATv2Conv}(\mathbf{H_1} + \mathbf{X}\mathbf{W}_{skip}, \mathbf{A})), \tag{5.3}$$

where $\mathbf{H_1}$ is the output of the previous layer, $\mathbf{X}$ are the input node features, $\mathbf{W}_{skip}$ is a learnable projection matrix for the skip connection, and $\mathbf{A}$ denotes the adjacency matrix.

Since large graphs, i. e., our proposed item co-occurrence graph, generally do not fit into GPU memory, we rely on a sub-graph sampling approach based on neighborhood batch sampling in [10], subsampling a fine-tuned, fixed-sized neighborhood per node.

5.3.2. Extension of Sequential Models

In sequential neural network models, next items are predicted by multiplying the candidate item embeddings with the learned session representation and applying the softmax operation to obtain the corresponding item probabilities. To combine the advantage of graph-based item embeddings and current state-of-the-art models in session-based recommendation, we propose the following approach: Instead of initializing the sequential model's item embedding table with weights based on sophisticated initialization methods (for instance, the widely used Xavier initialization [7]), we directly adopt the graph-generated item embeddings. Thereby, GCNext improves the learning process of the underlying sequential model as these embeddings capture topological knowledge about the item-item relations and contain additional item feature information.

We train the graph-based item embeddings in an unsupervised manner. Compared to end-to-end embedding learning, this has two advantages: First, GCNext is a modular method that can easily be applied to different sequential models without altering their architectures. Second, splitting up the training process into two stages supports the usage in production, since the training of sequential models with the pre-trained item embeddings converges faster.

In contrast to neural network models, nearest–neighbor methods do not make use of an item embedding table; they find similar sessions based on the input sequence to predict the next candidate item. To use GCNext in nearest–neighbor approaches, we propose using item graph embeddings to find similar session neighbors. To integrate item graph embeddings in nearest–neighbor methods, we compute the similarity of sessions based on the cosine distance of the embedding of each item in the input session to every item in the candidate session. A threshold value on the distance is adapted to find similar embeddings. Candidate sessions are then scored by the corresponding position mapping of the similar items based on the nearest–neighbor method or again by their cosine similarity; as defined by the r-score in Equation 5.4 [25].

$$\mathrm{r}(S^{(\mathrm{i})}, S^{(\mathrm{c})}) = \frac{|T_{\mathrm{i,c}}|}{\sqrt{|S^{(\mathrm{i})}|}\sqrt{|S^{(\mathrm{c})}|}}, \tag{5.4}$$

where $S^{(i)}$ and $S^{(c)}$ refer to the sets of items in input and candidate session, respectively, and $T_{i,c}$ represents the set of (pairs of) items in the input and candidate session with embedding similarity below the threshold.

5.4. Experimental Setup

5.4.1. Datasets and Preprocessing

To evaluate GCNext, we conduct experiments on three widely used datasets with different characteristics from the e-commerce and music domains. The **Diginetica**[1] dataset (CIKM Cup 2016) provides different item features; we use the category and price of each item as features (side information) [20, 21]. The **Tmall**[2] dataset as part of the IJCAI-15 competition contains users' shopping logs along with the category, brand, and seller as additional item features [36, 43]. Furthermore, we evaluate GCNext on the **Music4All+** dataset, a version of the Music4All[3] dataset [32] with 11 item features. We enrich this dataset with i-vectors [3] of dimensionality 100 based on the 13-dimensional Mel-Frequency Cepstral Coefficients of the songs and a Gaussian Mixture Model with 256 components [5], using the `kaldi` toolkit [29]. Similar to [32], we consider listening events to belong to the same session if there are no gaps of more than 30 minutes between them.

Following previous works [21, 41], we filter out items appearing less than 5 times and ignore sessions consisting of a single interaction. Additionally, training sequences are generated by splitting the input sequence into smaller sub-sequences. Consider, for example, the input sequence $s = [i_1, i_2, ..., i_n]$, then the generated sequences and corresponding next items are $([i_1], i_2), ([i_1, i_2], i_3), ..., ([i_1, i_2, ...i_{n-1}], i_n)$. The maximum sequence length is set to 50. Each dataset is sorted by its timestamps and temporally split into training, validation, and test set (80%, 10%, and 10%). Table 5.1 provides an overview of the datasets.

Table 5.1.: Dataset Statistics: Number of items, features, sessions, and average session length.

Dataset	Items	Feat.	Sessions	Avg. Length
Diginetica	44,527	2	205,698	4.85
Tmall	97,259	3	188,113	8.11
Music4All+	80,471	12	601,858	7.70

[1] https://cikm2016.cs.iupui.edu/cikm-cup/
[2] https://tianchi.aliyun.com/dataset/dataDetail?dataId=42
[3] https://sites.google.com/view/contact4music4all

5.4.2. Base Algorithms and Implementation

One of the main advantages of GCNext is that it can be added to sequential models in a plug-in fashion to boost their performance. Therefore, we evaluate the following base algorithms for session-based recommendation with and without adding GCNext: SKNN [16], STAN [6], V–SKNN [24], and VSTAN [25] as representative nearest–neighbor methods; GRU4Rec+ [34], Caser [36], NARM [20], STAMP [21], and SASRec [17] as state-of-the-art neural network models. In addition, we compare GCNext to current graph-based approaches (SR–GNN [41], GCSAN ,[42] and LightGCN [13]), and models including additional item features: GRU4RecF [15] and FDSA [43]. We additionally implement the graph-based approaches to incorporate the original item features in their initial embedding tables with naive sum-pooling, which are referred to as SR–GNNF, GCSANF, and LightGCNF.

All base models use the implementation in *RecBole* [44] for neural network methods and *session-rec* [25] for nearest-neighbor methods. For the BGRL implementation, we rely on the code given in [37]. We use AdamW [23] to optimize item graph embeddings and Adam [18] in the sequential model training. The embedding size is fixed to 128 for comparability and all models performed best with this configuration according to preliminary experiments. We conduct hyperparameter optimization via grid search including the learning rate, number of layers and heads, layer sizes, and dropout rates for augmentation. Each experiment is repeated five times and the average results are reported. We provide our implementation on Github[4].

We adopt two widely used evaluation metrics to assess the quality of the recommendation lists: *HR@k* (Hit Rate) and *MRR@k* (Mean Reciprocal Rank). Similar to previous works [4, 21, 31], each metric is computed with k set to 10 and 20.

5.5. Results and Analysis

Table 5.2 presents the experimental results of all base methods and their performance when extended with the proposed GCNext approach (column GCNext). When integrating GCNext into the nearest–neighbor methods (top part of Table 5.2), we observe a significant performance increase for some base models, and no significant decrease for any of the base models. In particular, V-SKNN's performance improves by 0.88% to 6.33% over all datasets on the *HR@10* score. Although the metrics show that nearest–neighbor methods are able to keep up with certain neural network approaches, they lack in state-of-the art performance across all three datasets.

The effectiveness of our approach is distinctly indicated by the results of neural network approaches in session-based recommendation. Compared to all neural network meth-

[4] https://github.com/dbis-uibk/gcnext

Table 5.2.: Model performances on all datasets; column GCNext indicates the use of GCNext. Significant improvements over the underlying sequential models (paired t-test, $p < .05$) are marked with †; best results in bold, second-best results underlined.

Model	GCNext	Diginetica				Tmall				Music4All+			
		MRR		HR		MRR		HR		MRR		HR	
		@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20
Nearest–neighbor Methods													
SKNN	✗	17.98	18.62	36.67	45.94	22.13	22.41	32.32	36.30	19.49	19.89	42.83	48.45
STAN	✗	15.43	16.12	28.95	38.97	20.13	20.41	26.15	30.20	18.48	18.85	37.56	42.78
V-SKNN	✗	17.99	18.63	36.59	45.84	21.84	22.09	30.95	34.55	19.58	19.99	43.06	48.68
VSTAN	✗	15.27	15.97	28.64	38.73	20.04	20.32	26.01	30.04	18.41	18.77	36.99	42.18
SKNN	✓	18.07†	18.71†	37.11†	46.40†	22.14	22.40	32.32	36.31	19.43	19.85	43.10†	48.90†
STAN	✓	15.43	16.12	28.95	38.97	20.14	20.42	26.16	30.21	18.49	18.85	37.60†	42.80†
V-SKNN	✓	18.18†	18.83†	37.23†	46.47†	22.38†	22.67†	32.91†	36.95†	19.66†	20.07†	43.44†	49.20†
VSTAN	✓	15.27	15.97	28.64	38.73	20.04	20.32	26.01	30.05	18.56†	18.92†	37.41†	42.57†
Neural Network Methods													
GRU4Rec+	✗	17.09	17.94	38.19	50.45	28.62	29.00	45.71	51.02	25.18	25.62	41.35	47.53
Caser	✗	14.29	14.82	26.54	34.20	24.30	24.54	35.61	39.04	19.26	19.62	31.51	36.63
STAMP	✗	16.38	17.17	35.62	47.04	21.63	21.88	31.62	35.21	28.12	28.45	41.96	46.60
NARM	✗	17.35	18.18	38.54	50.57	28.11	28.46	44.58	49.71	28.82	29.20	42.64	48.00
SASRec	✗	19.88	20.73	43.09	55.24	29.46	29.84	**47.72**	53.15	28.83	29.23	45.25	51.01
GRU4Rec+	✓	17.43†	18.27†	38.59†	50.78†	28.74†	29.12†	46.11†	51.60†	28.78†	29.18†	43.33†	48.99†
Caser	✓	15.63†	16.28†	30.60†	39.99†	26.33†	26.68†	41.15†	46.12†	20.56†	20.92†	33.35†	38.54†
STAMP	✓	17.43†	18.23†	37.50†	49.08†	24.33†	24.68†	37.75†	42.82†	28.60†	28.95†	42.81†	47.87†
NARM	✓	17.89†	18.73†	39.51†	51.73†	28.99†	29.39†	46.48†	52.19†	28.89†	29.28†	43.11†	48.67†
SASRec	✓	**19.97**†	**20.80**†	**43.10**	**55.41**†	**29.51**†	**29.93**†	47.70	**53.75**†	29.84†	30.15†	**45.43**†	**51.13**†
Feature & Graph-based Methods													
GRU4RecF	✗	16.04	16.91	36.35	48.77	25.25	25.66	42.04	47.97	28.67	29.04	42.52	47.80
FDSA	✗	18.92	19.79	41.29	53.73	28.76	29.15	46.48	52.01	30.04	30.43	45.20	50.66
SR-GNN	✗	17.75	18.58	39.18	51.23	27.47	27.84	44.67	49.89	28.90	29.27	42.62	47.91
SR-GNNF	✗	17.49	18.33	38.45	50.56	25.55	25.98	41.52	47.70	28.72	29.08	41.98	47.14
GCSAN	✗	19.20	20.03	41.00	53.01	29.01	29.41	47.37	53.04	29.68	30.05	43.79	49.09
GCSANF	✗	17.21	18.04	37.60	49.57	25.16	25.58	40.60	46.67	**30.13**	**30.49**	43.74	48.84
LightGCN	✗	15.90	16.67	35.11	46.29	26.03	26.47	45.92	52.16	8.43	9.01	21.98	30.18
LightGCNF	✗	15.90	16.66	34.96	45.92	25.80	26.27	45.49	52.25	10.95	11.59	28.49	37.45

ods, Caser, which uses convolution-based sequence embeddings, and STAMP, a complete attention-based method, benefit the most by incorporating GCNext across all three datasets. On the Diginetica dataset GCNext combined with SASRec, a transformer-based model, achieves the highest score on each metric throughout. This effect can also be seen with the Tmall dataset, where SASRec with GCNext increases the state-of-the-art *HR@10* score by 1.12% and significantly outperforms feature-based methods such as GRU4RecF and FDSA. STAMP extended with GCNext even achieves an increase in performance of 12.79% on the *MRR@20*. Interestingly, this effect becomes less impactful for the Music4All+ dataset which provides a large set of additional item features. On this dataset the graph-based self-attention network GCSAN extended with item features achieves the highest *MRR* scores. Nonetheless, the evaluation shows significantly improved scores for each of the neural network-based models on the Music4all+ dataset.

It is also noteworthy that neural network models trained with pre-trained item graph embeddings converge faster than their corresponding standard initialized counterparts. We assume this is due to the enriched information contained in the item embeddings.

5.6. Conclusion and Future Work

We proposed GCNext, an extension to sequential recommendation models based on GCNs. In the first phase, we generate an item co-occurrence graph with nodes/items enriched with item descriptors to learn its node representations in an unsupervised manner. In the second phase, we use the learned item embedding weights to initialize the item embedding table of the underlying sequential base model. Our experimental results on three different datasets show the effectiveness of GCNext.

For future work, we plan to further investigate the potential improvements of graph-based item embeddings in cold-start scenarios for session-based recommendation. Additionally, future experiments will compare our approach with different graph embeddings methods and investigate the diverging impact on baseline models.

References

[1] M. Amjadi, S. D. M. Taheri, and T. Tulabandhula. Katrec: knowledge aware attentive sequential recommendations. In *Discovery Science - 24th International Conference, DS 2021, Proceedings*, volume 12986 of *Lecture Notes in Computer Science*, pages 305–320. Springer, 2021.

[2] S. Brody, U. Alon, and E. Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations*, 2022. URL: https://openreview.net/forum?id=F72ximsx7C1.

[3] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798, 2010.

[4] G. de Souza Pereira Moreira, S. Rabhi, J. M. Lee, R. Ak, and E. Oldridge. Transformers4rec: bridging the gap between NLP and sequential / session-based recommendation. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems*, pages 143–153. ACM, 2021.

[5] H. Eghbal-zadeh, B. Lehner, M. Schedl, and G. Widmer. I-vectors for timbre-based music similarity and music artist classification. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015*, pages 554–560, 2015.

[6] D. Garg, P. Gupta, P. Malhotra, L. Vig, and G. Shroff. Sequence and time aware neighborhood for session-based recommendations: STAN. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019*, pages 1069–1072. ACM, 2019.

[7] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010*, volume 9 of *JMLR Proceedings*, pages 249–256. JMLR.org, 2010.

[8] A. Grover and J. Leskovec. Node2vec: scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 2016*, pages 855–864. ACM, 2016.

[9] T. R. Gwadabe and Y. Liu. Ic-gar: item co-occurrence graph augmented session-based recommendation. *Neural Computing and Applications*:1–16, 2022.

[10] W. L. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 1024–1034, 2017.

[11] K. Hassani and A. H. K. Ahmadi. Contrastive multi-view representation learning on graphs. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, volume 119 of *Proceedings of Machine Learning Research*, pages 4116–4126. PMLR, 2020.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision, ICCV 2015*, pages 1026–1034. IEEE Computer Society, 2015.

[13] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang. Lightgcn: simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, pages 639–648. ACM, 2020.

[14] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, Conference Track Proceedings*, 2016.

[15] B. Hidasi, M. Quadrana, A. Karatzoglou, and D. Tikk. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems 2016*, pages 241–248. ACM, 2016.

[16] D. Jannach and M. Ludewig. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017*, pages 306–310. ACM, 2017.

[17] W. Kang and J. J. McAuley. Self-attentive sequential recommendation. In *IEEE International Conference on Data Mining, ICDM 2018*, pages 197–206. IEEE Computer Society, 2018.

[18] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*, 2015.

[19] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.

[20] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017*, pages 1419–1428. ACM, 2017.

[21] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018*, pages 1831–1839. ACM, 2018.

[22] Z. Liu, Y. Ma, M. Schubert, Y. Ouyang, and Z. Xiong. Multi-modal contrastive pre-training for recommendation. In *ICMR '22: International Conference on Multimedia Retrieval*, pages 99–108. ACM, 2022.

[23] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[24] M. Ludewig, I. Kamehkhosh, N. Landia, and D. Jannach. Effective nearest-neighbor music recommendations. In *Proceedings of the ACM Recommender Systems Challenge, RecSys Challenge 2018*, 3:1–3:6. ACM, 2018.

[25] M. Ludewig, N. Mauro, S. Latifi, and D. Jannach. Empirical analysis of session-based recommendation algorithms. *User Modeling and User-Adapted Interaction*, 31(1):149–181, 2021.

[26] Z. Meng, S. Liu, C. Macdonald, and I. Ounis. Graph neural pre-training for enhancing recommendations using side information. *CoRR*, abs/2107.03936, 2021.

[27] Z. Pan, W. Chen, and H. Chen. Dynamic graph learning for session-based recommendation. *Mathematics*, 9(12), 2021. ISSN: 2227-7390.

[28] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: online learning of social representations. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14*, pages 701–710. ACM, 2014.

[29] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely. The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011.

[30] M. Quadrana, P. Cremonesi, and D. Jannach. Sequence-aware recommender systems. *ACM Computing Surveys (CSUR)*, 51(4):1–36, 2018.

[31] P. Ren, Z. Chen, J. Li, Z. Ren, J. Ma, and M. de Rijke. Repeatnet: A repeat aware neural recommendation machine for session-based recommendation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 4806–4813. AAAI Press, 2019.

[32] I. A. P. Santana, F. Pinhelli, J. Donini, L. G. Catharin, R. B. Mangolin, Y. M. e Gomes da Costa, V. D. Feltrim, and M. A. Domingues. Music4all: A new music database and its applications. In *2020 International Conference on Systems, Signals and Image Processing, IWSSIP 2020*, pages 399–404. IEEE, IEEE, 2020.

[33] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang. Bert4rec: sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019*, pages 1441–1450. ACM, 2019.

[34] Y. K. Tan, X. Xu, and Y. Liu. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS@RecSys 2016*, pages 17–22. ACM, 2016.

[35] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. LINE: large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015*, pages 1067–1077. ACM, 2015.

[36] J. Tang and K. Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018*, pages 565–573. ACM, 2018.

[37] S. Thakoor, C. Tallec, M. G. Azar, R. Munos, P. Veličković, and M. Valko. Bootstrapped representation learning on graphs. In *ICLR 2021 Workshop on Geometrical and Topological Representation Learning*, 2021.

[38] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[39] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm. Deep graph infomax. In *7th International Conference on Learning Representations, ICLR 2019*. OpenReview.net, 2019.

[40] B. Wang and W. Cai. Knowledge-enhanced graph neural networks for sequential recommendation. *Inf.*, 11(8):388, 2020.

[41] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan. Session-based recommendation with graph neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, volume 33 of number 01, pages 346–353. AAAI Press, July 2019.

[42] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, and X. Zhou. Graph contextualized self-attention network for session-based recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 3940–3946. ijcai.org, 2019.

[43] T. Zhang, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, D. Wang, G. Liu, and X. Zhou. Feature-level deeper self-attention network for sequential recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 4320–4326. ijcai.org, 2019.

[44] W. X. Zhao, S. Mu, Y. Hou, Z. Lin, Y. Chen, X. Pan, K. Li, Y. Lu, H. Wang, C. Tian, Y. Min, Z. Feng, X. Fan, X. Chen, P. Wang, W. Ji, Y. Li, X. Wang, and J. Wen. Recbole: towards a unified, comprehensive and efficient framework for recommendation algorithms. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*, pages 4653–4664. ACM, 2021.

6. Efficient Session-based Recommendation with Contrastive Graph-based Shortest Path Search

Publications

A. Peintner, A. R. Mohammadi, and E. Zangerle. SPARE: shortest path global item relations for efficient session-based recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, pages 58–69. ACM, 2023. DOI: 10.1145/3604915.3608768

Extended version:
A. Peintner, A. R. Mohammadi, and E. Zangerle. Efficient session-based recommendation with contrastive graph-based shortest path search. *ACM Transactions on Recommender Systems*, 3(4), Apr. 2025. DOI: 10.1145/3701764

Abstract

Session-based recommendation aims to predict the next item based on a set of anonymous sessions. Capturing user intent from a short interaction sequence imposes a variety of challenges since no user profiles are available and interaction data is naturally sparse. Recent approaches relying on graph neural networks (GNNs) for session-based recommendation use global item relations to explore collaborative information from different sessions. These methods capture the topological structure of the graph and rely on multi-hop information aggregation in GNNs to exchange information along edges. Consequently, graph-based models suffer from noisy item relations in the training data and introduce high complexity for large item catalogs. We propose to explicitly model the multi-hop information aggregation mechanism over multiple layers via shortest-path edges based on knowledge from the sequential recommendation domain. Our approach does not require multiple layers to exchange information and ignores unreliable item-item relations. Furthermore, to address inherent data sparsity, we are the first to apply supervised contrastive learning by mining data-driven positive and hard negative item samples from the training data. Extensive experiments on four different datasets show that the proposed approach outperforms almost all of the state-of-the-art methods.

6.1. Introduction

Recommender systems are an important tool for users to obtain useful information. They are widely adopted in various areas like e-commerce or online streaming services and implicitly boost business revenue by improving user experience. However, most conventional recommender systems rely on the availability of user profiles and long-term interaction histories and therefore, are not suitable for scenarios in which this data is not available (for instance, anonymous sessions). Tackling this task, session-based recommendation (SBR) aims at predicting the next most likely item based solely on an anonymous session [26].

Early works in this field considered Markov Chains and recurrent neural networks (RNNs) to model the temporal dependencies of items in the session sequence [10, 27]. Based on the similarity of sessions, nearest-neighbor methods were also deployed for session-based recommendation [5, 12, 20]. Other approaches incorporated convolutional neural networks [31, 45] and attention mechanisms [17, 19]. Recent studies have deployed graph neural networks (GNNs) to model sessions via graphs and have been shown to be state-of-the-art [8, 18, 34, 37, 38, 39, 40]. However, the success of current GNN models relies on using complex multi-layer graphs [8, 34] or several graphs to augment different aspects of data [38, 39]. While these approaches complement collaborative information, they can also introduce irrelevant information that adversely affects recommendation performance as well as being inefficient and computationally expensive [48]. On the other hand, generating different views of a graph via augmentation with different edge drop-out rates, for instance, does not adversely affect the performance of contrastive learning-based recommendation models, and in fact, even large drop-out rates on edges (e.g., 0.9) are beneficial [43]. Considering these two findings, we investigate the more general questions: How can we leverage de-noised, simpler graphs for SBR and how do they compare to complex, noisy graphs?[1]

Taking into account the above-discussed limitations of noisy and computationally expensive input graphs, in this paper, we propose **S**hortest-**Pa**th **Re**lations (**SPARE**) to enrich a global item graph with informative connections. With SPARE, we introduce a graph-building strategy that relies on a shortest-path search to drop irrelevant item connections in the graph. This procedure can be considered as edge sparsification in the graph and is correlated with the long-standing concept of finding frequent item sets with high support [1]. As a further important benefit, adding shortest-path shortcut connections explicitly models item-item importance and imitates the n-hop neighbor information aggregation of standard GNNs with multiple layers for efficient item representation learning. We illustrate this concept in Figure 6.1, where we present template sessions of an e-commerce grocery retailer. In this example, we have dough, salami, tomato, and cheese—ingredients in a pizza recipe—in our frequently occurring sessions

[1] Please note that this manuscript is an extended version of [24], which was presented at the 17th ACM Conference on Recommender Systems (RecSys 2023).

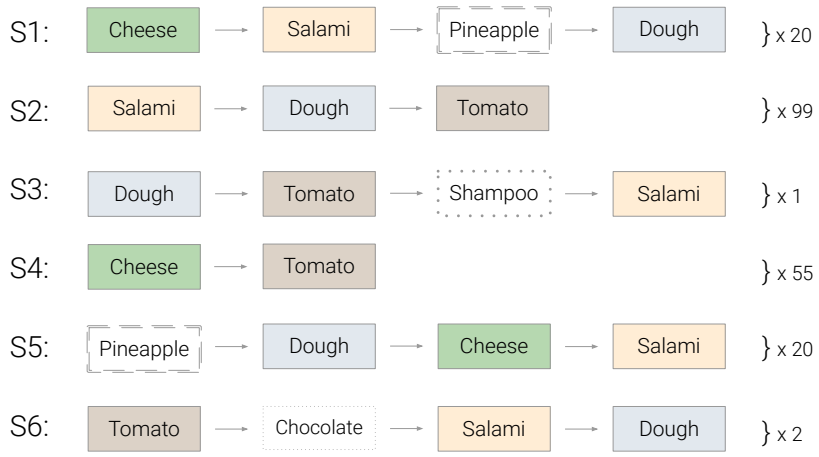


Figure 6.1.: A toy example of an e-commerce grocery retailer scenario. Numbers indicate the frequency of each session.

(sessions 2 and 4). There are also people who purchase less-common ingredients such as pineapple amongst pizza ingredients (sessions 1 and 5). Additionally, in some sessions, customers may buy unrelated items, such as shampoo or chocolate (sessions 3 and 6). The purchase of shampoo or chocolate seems like an irrelevant outlier for a customer who is looking for ingredients for a pizza recipe. However, pineapple should be considered as an interesting pattern for the customers who buy pizza ingredients, even in the case that tomato is in the basket (no co-purchase). Through the high support of pineapple -> dough and dough -> tomato relations, a shortest-path search in a global item graph finds a direct connection (shortcut connection) between pineapple and tomato. Furthermore, since item relations containing shampoo or chocolate have low support, shortest-path search disregards them as irrelevant, given a proper threshold value, resulting in a sparse global item graph. However, graph edge sparsification comes with the risk of increasing data sparsity and popularity bias. To counteract the sparsity of data and reinforcement of the popular item sets, for the first time, we apply Supervised Contrastive Learning (SCL) [14] by mining positive and negative item samples in a data-driven manner. With SCL, we not only tackle the sparsity of data but also improve the model's performance by refining the encoder and item representations through the self-supervised learning objective.

We summarize our main technical contributions as follows:

- We propose a novel global item graph-building strategy (SPARE) based on shortest paths to introduce item shortcut connections and graph edge sparsification.

- We integrate a supervised contrastive learning task based on data-driven hard negative samples to tackle data sparsity and the inherent popularity bias to enhance recommendation performance.

- Extensive experiments show that our proposed model provides higher efficiency while significantly outperforming state-of-the-art competitors.

- To ensure reproducibility, we published the code of our experiments and analysis at GitHub[2].

6.2. Related Work

Sequential recommendation leverages user data and long-term interactions, whereas session-based recommendation is limited to anonymous sessions only. In this section, we review both tasks and present related research.

6.2.1. Sequential Recommendation

The initial phase of sequential recommendation focuses on discovering short-term item representations and interaction patterns. Markov decision processes are used in early works to model the interaction sequences. In FPMC [27], first-order Markov chains capture sequential patterns while matrix factorization models long-term user preferences. Also, convolutional neural networks (CNNs) have been found to be useful, where items are seen as images and short-term sequential patterns are learned via convolutional filters [30]. Xu et al. [41] combine CNNs with long-short-term memory to extract additional complex long-term dependencies. In HGN [21], a feature and instance gating mechanism is used to capture long- and short-term user interests. Other studies apply the attention mechanism to obtain and fuse different levels of interaction information [29, 44].

Self-attention and Transformer-based architectures are widely used for sequential recommendation models. SASRec [13] applies the self-attention mechanism to identify relevant interactions from the user's history. Others use custom Transformer models to provide more personalized recommendation [3, 36]. In FDSA [49], heterogeneous features of items are integrated via feature sequences, and self-attention is applied to jointly model item and feature transition patterns. S^3-Rec [51] utilizes self-supervised learning to enhance the item representations via pre-training methods.

Hsu and Li [11] extract a local subgraph from a user-item pair and apply self-attention to encode long-term and short-term temporal patterns. MA-GNN [22] captures the item contextual information within a short-term period with a graph neural network

[2] https://github.com/dbis-uibk/SPARE

and utilizes a shared memory network to model long-range dependencies. Zhang et al. [50] extract augmented sequences representations from an item transition graph for a contrastive learning objective.

6.2.2. Session-based Recommendation

In session-based recommendation, user profiles and long-term interaction histories are no longer available. Consequently, the goal is to effectively model informative session representations. Early works adopted recurrent neural networks (RNNs) to model the sequentiality of item interactions. GRU4Rec [10] uses gated recurrent units (GRUs) to encode interaction sequences. This approach is extended in NARM [17] with an attention mechanism that additionally captures the main intent of a session. To capture the general interest based on the long-term interaction history and the current interest from the most recent clicks, STAMP [19] introduces a short-term attention/memory priority model.

Based on the knowledge contained in other sessions, a different line of research extracts collaborative information for improved session representations. SKNN [12] finds sessions containing the same elements as the current session and relies on selecting items from the most similar session. Its successor VSKNN [20] extends this approach by taking the position and frequency of items into account. Another nearest-neighbor approach named STAN [5] additionally incorporates factors like recency and different item position weighting strategies. In CSRM [33], neighborhood sessions are used to extract collaborative information in a hybrid framework with two parallel memory modules.

Most recent works in session-based recommendation are based on GNNs. As the first to introduce the concept of representing sessions as graphs, SR-GNN [37] models each session as a directed, unweighted graph and applies a gating mechanism to generate session representations. This work is extended by a self-attention mechanism in GCSAN [40] to effectively capture long-range dependencies. Incorporating collaborative knowledge into GNN-based methods leads to a new line of research. GCE-GNN [34] learns item embeddings on a session level as well as on a global level and uses a soft-attention mechanism to fuse the learned item representations. Xia et al. [38] introduce a dual-channel hypergraph to capture beyond-pairwise relations and apply self-supervised learning to maximize the mutual information between both session representations. MGIR [8] shows that global incompatible items are informative and aggregate positive and negative relations for the final session representation. In DGNN [18] a dual graph neural network models explicit dependencies among items and employs a self-learning strategy to capture implicit correlations among items.

However, some works investigate the limits of using the GNN framework to capture pairwise relationships among items. Work in [48] proposes to remove redundant modules and to focus more on the readout module to achieve multi-level reasoning over item transitions. Chen and Wong [2] tackle the long-range dependency (over-smoothing) problem of

session graphs by introducing a lossless encoding scheme and a shortcut graph attention layer. Yang et al. with SPAGAT [42] are the first to introduce the concept of shortest-path attention in GNNs by applying a complex path feature aggregation strategy and is therefore not feasible for recommender systems.

With this work, we are the first to exploit shortest-path search to introduce shortcut connections in a global item graph which significantly increases the computational efficiency of the model. Also, compared to other self-supervised learning methods tackling the data sparsity in SBR, our approach is the first to mine supervised positive and hard negative item samples for the computation of the contrastive loss.

6.3. Preliminaries

In this section, we first introduce the problem statement and important notations for session-based recommendation. Subsequently, we present the construction of the global item base graph which is based on the sequential appearances of item interactions in the session data.

6.3.1. Problem Statement and Notations

Let $\mathcal{I} = \{i_1, i_2, i_3, ..., i_N\}$ be the item universe, where N is the number of items. Each session consists of sequential, temporally ordered interactions with items and is denoted by $s = [i_1^s, i_2^s, i_3^s, ..., i_l^s]$, where l is the length of session s and i_j^s represents the j^{th} item interacted with within this session. Item representations are learned by encoding all items $i \in \mathcal{I}$ into the same embedding space. Using d dimensions for the embedding, the item representation set is denoted as $\mathbf{X} \in \mathbb{R}^{N \times d}$ and is randomly initialized with a uniform distribution. Given a session s, the task of session-based recommendation is to predict the next item i_{l+1}^s of the interaction sequence.

6.3.2. Global Item Base Graph

To capture all item relations in the sessions, a global item graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is constructed. This weighted directed graph is defined by $\mathcal{V} = \mathcal{I}$ being the item catalog set and $\mathcal{E} = \{\varepsilon_{ij}\}$ the set of all sequential relations between items. There exists an edge ε_{ij} from node v_i to node v_j if item i_i is being directly followed by item i_j in a session. Each edge ε_{ij} is assigned a weight w_{ij} defined by the frequency of consecutive appearances of both items across all sessions. This global item base graph is by nature sparse since items are usually connected to a very small subset of other items based on the context of a session.

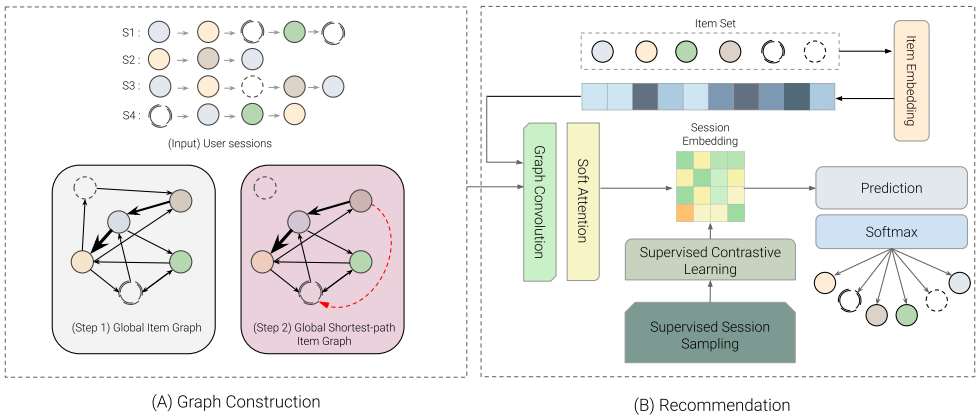


Figure 6.2.: An overview of the global item graph construction and the pipeline of the proposed SPARE model. During the graph construction SPARE builds a global item graph from all user interactions as a first step. In the second step of the graph construction, the shortest-path search induces shortcut connections in the graph (red arrows) and drops noisy edges. In the recommendation module, SPARE utilizes graph convolutions and soft attention to learn session embeddings which are enriched by supervised contrastive learning through sampling data-driven positive and negative sessions based on a custom distance metric.

6.4. Proposed Method

In this section, we present the proposed **S**hortest-**Pa**th **Re**lations (**SPARE**) global item graph and the proposed supervised contrastive learning approach for efficient session-based recommendation based on the SPARE graph. Figure 6.2 presents an overview of the components in SPARE. First, the global base item graph is enriched by shortest-path connections in the graph construction phase. The resulting graph is input to the recommendation component. Particularly, to our graph convolutional layer leading to learned session representations enhanced by the supervised contrastive learning task for SBR. Each component will be described in detail in the following.

6.4.1. Sparse and Shortest-Path Aware Item Graph

Most graph-based models in SBR using global item graphs rely on $\mathcal{G}$ as their workhorse which by design tends to be noisy and only contains sequential relations of items. Most existing models based on GNNs for SBR cannot capture long-range dependencies (items that are multiple hops apart), since they are limited by the receptive field of each node per layer (1-hop neighbors). Stacking multiple GNN layers enables them to capture multi-hop relations, but introduces the problem of over-smoothing (node representations converge to the same value) if the number of layers is larger than three [2, 15]. However, in real-world datasets, it is very common that sessions contain more than three item

interactions; yet, items separated over longer distances hold valuable information (cf. also the dataset statistics presented in Table 6.1). To solve this issue, we introduce the concept of finding shortest paths in the global item graph to insert suitable shortcut connections between items and circumvent the problem of over-smoothing.

There exist many efficient algorithms to find the shortest paths between two nodes in a given graph. In this work, we rely on the widely used Dijkstra algorithm using Fibonacci Heaps [4] due to its low computational cost. We transform each edge weight to its inverse weight by subtracting its weight from the maximal weight of all edges to get the corresponding cost c_{ij} to get from node v_i to node v_j. Then, for each node in the global item graph $\mathcal{G}$, the shortest path to every other node is computed based on the minimal cost of the sum of edge costs in the path. The receptive field of each node and the sparsity of the graph is controlled via the μ limit parameter. Choosing μ to be in an acceptable range serves as a threshold value to filter out relations not being sufficiently supported in the graph, tackling the problem of noisy sequences introduced in the training data which can mislead the model as shown in [8]. The edge costs $\hat{c}_{ij}$ found through the shortest-path search and the final edge weights $\hat{w}_{ij}$ in the resulting graph $\hat{\mathcal{G}}$ are defined as:

$$\delta_{ij} = \sum_{i=1}^{n-1} c_{i,i+1} \tag{6.1}$$

$$\hat{c}_{ij} = \begin{cases} \delta_{ij}, & \text{if } \delta_{ij} \leq \mu \\ 0, & \text{otherwise} \end{cases} \tag{6.2}$$

$$\hat{w}_{ij} = max(\hat{\mathbf{C}}) - \hat{c}_{ij}, \tag{6.3}$$

where the sum of individual edge costs δ_{ij} is minimized by path $P = \{v_i, v_{i+1}, ..., v_j\}$ with length n over all possible nodes and $\hat{\mathbf{C}} \in \mathbb{R}^{N \times N}$ is the final cost matrix where each entry $\hat{c}_{ij}$ corresponds to the minimum cost going from node v_i to node v_j. Additionally, with this approach, we are able to include non-direct relations from the original graph $\mathcal{G}$ as shortcut connections with an adapted weight based on the hop distance. We hypothesize that these weighted shortcut connections imitate the n-hop neighbor information aggregation of standard GNNs with multiple layers, explicitly modeling item-item importance.

Compared to [2] which introduces a local, unweighted graph representation per session and therefore, also includes misleading item connections, our approach is able to filter out noisy item-item relations globally and models the importance of items effectively via corresponding edge weights.

6.4.2. Path-based Session Graph Encoder

The proposed shortest-path-aware global item graph now contains reliable pairwise item transitions from all sessions. We use a simple graph convolution to encode connections in the graph:

$$\mathbf{H} = \hat{\mathbf{D}}^{-\frac{1}{2}} \hat{\mathbf{A}} \mathbf{X} \hat{\mathbf{D}}^{-\frac{1}{2}}, \tag{6.4}$$

with $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, where $\mathbf{A}$ denotes the adjacency matrix $\mathbf{A} \in \mathbf{R}^{N \times N}$, $\mathbf{I}$ the identity matrix and $\mathbf{X} \in \mathbb{R}^{N \times d}$ are the initial item embeddings. We symmetrically normalize the adjacency matrix $\hat{\mathbf{A}}$ by its degree matrix $\hat{\mathbf{D}}$. As shown in [35, 38], applying a non-linear activation function is not essential for recommender systems and is therefore neglected in this convolutional operation.

In contrast to previous approaches [8, 34], our model does not make use of an attention mechanism to learn the importance of different neighbors but directly adopts the edge weights to quantify the importance of neighboring nodes. We argue that this non-parametric data-driven design more efficiently makes use of the shortest-path adjacency matrix, where each item-item connection already has a corresponding weight, reflecting the importance based on sequences in the data. Since our global item graph also contains shortcut connections to nodes that are multiple hops away, our approach only requires a single convolutional layer (in contrast to other methods that require multiple layers to increase the size of the receptive field per node). We investigate the impact of this design on efficiency in Section 6.5.6.

After performing the graph convolutional operation we obtain the global item graph representations for each item in a session s, e. g., $\mathbf{H}_s = [\mathbf{h}_{v_1^s}, \mathbf{h}_{v_2^s}, ..., \mathbf{h}_{v_l^s}]$.

Following [8, 34, 38], we model the sequentiality in sessions via reversed position embeddings. Due to the fact that sessions are of different lengths, reversed position embeddings are able to capture the item importance based on the position in the session more effectively. The learnable position embedding matrix $P = [p_1, p_2, p_3, ..., p_l]$, where l is the length of the current session and p_i represents the embedding vector for position i, is integrated into the item representation via concatenation and non-linear transformation:

$$\mathbf{h}'_\mathbf{i} = \tanh\left(\mathbf{W}_1 \left[\mathbf{h}_{v_i^s} || \mathbf{p}_{l-i+1}\right] + \mathbf{b}_1\right), \tag{6.5}$$

where $\mathbf{W_1} \in \mathbb{R}^{d \times 2d}$ and $\mathbf{b_1} \in \mathbb{R}^d$ are learnable parameters.

Session embeddings are computed by aggregating the item representations contained in the session. To further refine the session embeddings, a soft attention mechanism is usually applied in graph-based SBR models to prioritize different items in the session [34, 38]. By using this technique, attention weights are obtained as follows:

$$\alpha_i = \mathbf{q}^\intercal \sigma\left(\mathbf{W}_2 \mathbf{h}'_\mathbf{i} + \mathbf{W_3} \mathbf{h_s} + \mathbf{b}_2\right), \tag{6.6}$$

where $\mathbf{W_2}, \mathbf{W_3} \in \mathbb{R}^{d \times d}$ and $\mathbf{q}, \mathbf{b_2} \in \mathbb{R}^d$ are trainable parameters. The average of the session's item representations is denoted by $\mathbf{h_s}$. The final session representation is obtained via linear combination:

$$\mathbf{z} = \sum_{i=1}^{l} \alpha_i \mathbf{h}_{v_i^s} \tag{6.7}$$

6.4.3. Supervised Contrastive Learning

Contrastive learning, particularly in a self-supervised framework, is often employed in SBR to mitigate inherent challenges such as popularity bias and data sparsity, which can lead to trivial solutions. Previous works employing self-supervised learning for SBR [38, 39] use different views of a single session as ground truth (positive) supervision signals and views from other sessions in the mini-batch as negative. In this scenario, InfoNCE [32] has proven to be a successful learning objective [38, 39]. However, previous approaches fully neglect the available label information for sampling positive and negative samples leading to noisy class representations [14]. In our approach, we explicitly mine data-driven positive and hard negative item samples from all training sessions. The selection of hard negative item samples is crucial to truly contribute to the gradient of the optimization.

For mining data-driven item samples we define positive sessions as sessions in the training data with the same target item as the input session. Based on the assumption that in session-based scenarios the last-clicked item in a session is most important to the target item, the last items in each of the positive sessions and the target item of the input session are seen as positive item samples. To ensure the same amount of positive samples per session in a batch, k positive sessions are randomly sampled from all available positive sessions per input session s resulting in $c_k^{s^+}$.

To find hard negative items, all sessions containing one or more items from the input session, excluding sessions with the same target item, are sampled. These negative candidate sessions are refined by borrowing a metric from the NLP domain: To our best knowledge, we are the first to use the BLEU score [23] for session similarity computation. In contrast to nearest-neighbor methods for SBR which rely mainly on set-based similarity measures [12, 20], the BLEU score is easily applicable in sequential scenarios. It relies on a modified precision score p_n for n-grams up to length N which we adopt to the setting of SBR: We count the number of matching n-grams of items between reference sessions (input and positive) and each of the negative session candidates. Then the candidate counts are summed up and normalized. With this modification, repeated item appearances are penalized, allowing for more informative negative session candidates.

BLEU essentially computes the geometric average of the n-grams precision and additionally adds a brevity penalty (BP):

$$BP = \begin{cases} 1 & \text{if } l_n > l_p \\ e^{(1-l_p/l_n)} & \text{if } l_n \leq l_p, \end{cases} \tag{6.8}$$

where l_n is the session length (number of interacted items) of the negative candidate and l_p is the session length of the input or the positive sample session closer to l_n. The BP favors sessions with the exact same length as the reference sessions and prevents too short/long sessions from being selected as a hard negative session sample. Given this brevity penalty BP, the BLEU score is computed as follows:

$$BLEU = BP \cdot exp\left(\sum_{n=1}^{N} w_n \log \rho_n\right), \tag{6.9}$$

where w_n are positive, uniform weights (e. g., $1/N$) to compute the geometric mean of different n-gram sizes and ρ_n are the n-grams precision scores. We use the top-k sessions with the highest BLEU score (denoted as S^-_{BLEU}) as hard negative sessions and use their last-clicked items and target items as negative item samples $c_k^{s^-}$:

$$c_k^{s^-} = \text{top-}k\left(S^-_{BLEU}\right). \tag{6.10}$$

Following InfoNCE [32, 38] to maximize the agreement between the representations of the last-clicked items and the target items in combination with the session context, the learning objective is defined as follows:

$$\mathcal{L}_{SCL} = -\log \frac{\sum_{i \in c_k^{s^+}} \psi\left(\mathbf{h}_s^{last}, \mathbf{z}_s, \mathbf{h}^i\right)}{\sum_{i \in c_k^{s^+}} \psi\left(\mathbf{h}_s^{last}, \mathbf{z}_s, \mathbf{h}^i\right) + \sum_{j \in c_k^{s^-}} \psi\left(\mathbf{h}_s^{last}, \mathbf{z}_s, \mathbf{h}^j\right)}, \tag{6.11}$$

where $\mathbf{h}_s^{last}$ is the graph representation of the last-clicked item of the given input session s and $\psi(x_1, x_2, x_3)$ is defined as $exp(f_D(x_1 + x_2,\, x_2 + x_3)$ with temperature parameter τ to control the effect of discrimination. The discriminator function $f_D(\cdot)$ takes two vectors as input and scores the agreement between them. In our case, we implemented the cosine operation as discriminator. This contrastive learning approach refines the representations of the last-clicked items and the target item so that the model is able to distinguish between positive sessions and similar, but different target item sessions more effectively. Since this self-supervised loss incorporates target information from the training data to contrast positive and negative samples, this learning approach can be regarded as supervised contrastive learning.

6.4.4. Prediction and Model Optimization

Based on the learned item and session representations, the final score for each candidate item $v_i \in \mathcal{V}$ to be recommended for a session is computed by the dot product of the

session representation and the global item graph representations. We use a weighted normalization [7, 46] which has been shown to improve the training process stability and sensitivity to hyper-parameters:

$$\hat{\mathbf{z}} = w_z \mathrm{L}_2 Norm(\mathbf{z}), \hat{\mathbf{h}}_i = \mathrm{L}_2 Norm(\mathbf{h}_i) \tag{6.12}$$

$$y_i = \hat{\mathbf{z}}^\intercal \hat{\mathbf{h}}_i, \tag{6.13}$$

where w_z is the normalized weight, $\mathbf{z}$ corresponds to the final session representation and $\mathbf{h}_i$ is the computed global item graph embedding of item i. $\mathrm{L}_2 Norm$ denotes the L_2 normalization function.

The final prediction probabilities $\hat{y}_i$ are computed by applying the softmax function to the score of each candidate item:

$$\hat{y}_i = \frac{exp(y_i)}{\sum_{v_j \in \mathcal{V}} exp(y_j)}. \tag{6.14}$$

As a loss function to be minimized, the cross-entropy of the prediction results $\hat{y}$ is used:

$$\mathcal{L}_{CE}(y, \hat{y}) = -\sum_{i}^{|\mathcal{V}|} (y_i \, log \, (\hat{y}_i) + (1 - y_i) \, log \, (1 - \hat{y}_i)) + \lambda ||\Theta||_2^2, \tag{6.15}$$

where y denotes the one-hot encoding vector of the ground truth item. Additionally, λ is a hyper-parameter to control the L_2 regularization, given Θ as the model parameters.

For the final loss, we combine the recommendation task with the supervised contrastive learning task, where the total loss is given as:

$$\mathcal{L} = \mathcal{L}_{CE} + \beta \mathcal{L}_{SCL}, \tag{6.16}$$

where β is a hyper-parameter to control the magnitude of the contrastive learning. This loss is then jointly optimized during training. The whole training procedure of the proposed SPARE model is summarized in Algorithm 1.

6.5. Experiments and Results

In this section, we provide the setup and results of extensive experiments to evaluate our proposed SPARE model, where we compare SPARE to various state-of-the-art models in SBR. We establish the following research questions to investigate the impact of the proposed graph edge modifications and contrastive learning approach and whether contrastive learning-based approaches indeed require complex graph structures:

- **RQ1:** How does SPARE perform compared to other state-of-the-art SBR methods on different datasets?

Algorithm 1: Training procedure of SPARE.

Input : Training sessions S, item embeddings $\mathbf{X}$
Output: Recommendation list per session

```
Construct global item graph 𝒢;
Compute shortest-path global item graph Ĝ given threshold parameter μ;
foreach epoch do
    foreach batch do
        Learn global item graph representations through Eq. (6.4);
        foreach session s do
            Compute session representation following Eq. (6.5) to Eq. (6.7);
            Obtain positive and negative item samples via Eq. (6.8) to Eq. (6.10);
            Compute supervised contrastive learning loss with Eq. (6.11);
        end
        Jointly optimize the supervised and self-supervised objectives in Eq. (6.16);
    end
end
```

- **RQ2:** How do different components in SPARE contribute to the performance?
- **RQ3:** How sensitive is SPARE to different settings of hyperparameters (e. g., μ, w_z, k)?
- **RQ4:** How does SPARE perform under different similarity measures for computing the contrastive samples?
- **RQ5:** What is the impact of SPARE in terms of efficiency compared to other graph-based models?
- **RQ6:** How does SPARE perform with sessions of different length?
- **RQ7:** To which extent can the integration of SPARE's graph-building strategy boost the performance of other recommender models?
- **RQ8:** How does SPARE alter the graph structure compared to other baselines?

6.5.1. Experimental Setup

Datasets and Preprocessing

To evaluate the performance of our approach, we conduct experiments on four representative and widely used datasets from the e-commerce and music domains. The *Tmall*[3] dataset was published as part of the IJCAI-15 competition and contains user logs of an online shopping platform. *RetailRocket*[4] is a dataset on user browsing activities within six months and was released by an e-commerce company as part of a Kaggle contest. Next, the *Last.fm*[5] dataset comes from the music domain and includes music listening histories in which items are artists of the listened songs. Lastly, *Gowalla*[6] is a location check-in dataset and widely used for point-of-interest recommendation.

Table 6.1.: Dataset statistics: Number of sessions, unique items and average session length (after preprocessing).

Dataset	**# Train**	**# Test**	**# Items**	**Avg. Length**
Tmall	351,268	25,898	40,728	6.69
RetailRocket	433,643	15,132	36,968	5.43
Last.fm	2,837,330	672,833	38,615	11.78
Gowalla	675,561	155,332	29,510	3.85

We follow the preprocessing steps used in [34, 37] for the four datasets. To be more specific, sessions with length 1 and items appearing less than 5 times are filtered out across all datasets. The most recent data (e. g., last week) is set as test data and the remaining data serves as training data. Additionally, we augment a session $S = [i_1, i_2, ..., i_n]$ with a sequence splitting method which leads to multiple labeled sequences $([i_1], i_2), ([i_1, i_2], i_3), ..., ([i_1, i_2, ...i_{n-1}], i_n)$, where the last item in each set is the corresponding label (or target item) of the sequence. Additionally for *Gowalla* we follow previous works [2, 6] and keep the top 30,000 most popular locations and generate sessions by grouping user check-in records per day. Table 6.1 provides an overview of the datasets after preprocessing.

Evaluation Metrics

Following previous works [34, 38, 47], we adopt *P@k* (Precision) and *MRR@k* (Mean Reciprocal Rank) to evaluate the quality of the recommendation results. For each metric, k is set to 10 and 20.

[3] https://tianchi.aliyun.com/dataset/dataDetail?dataId=42
[4] https://www.kaggle.com/retailrocket/ecommerce-dataset
[5] http://ocelma.net/MusicRecommendationDataset/lastfm-1K.html
[6] https://snap.stanford.edu/data/loc-gowalla.html

Baseline Methods

We compare SPARE with the following representative baseline and state-of-the-art methods for session-based recommendation:

- **Item-KNN** [28]: recommends items based on the similarity between items of the current session and the items of other user sessions.
- **FPMC** [27]: captures sequential effects and user preferences with matrix factorization and first-order Markov chains. To make it applicable for session-based recommendation, user latent representations are not used when computing recommendation scores.
- **GRU4Rec** [10]: a RNN-based method that applies Gated Recurrent Unit (GRU) in combination with a ranking-based loss function to model user interaction sequences.
- **NARM** [17]: extends GRU4Rec with an attention mechanism to capture the user's main purpose efficiently.
- **STAMP** [19]: replaces all RNN encoders in previous works by attention layers and relies on the self-attention mechanism of the last item to capture short-term interests.
- **SR-GNN** [37]: employs a gated GCN layer to obtain item embeddings. Similarly to STAMP, self-attention of the last item is used to compute the session embeddings.
- **FGNN** [25]: converts sessions into directed graphs and uses a graph attention layer to learn item representations.
- **GCE-GNN** [34]: constructs a session-level and a global co-occurrence graph to capture local and global information of items.
- S^2**-DHCN** [39]: captures beyond pairwise-relations with hypergraph modeling. It additionally integrates self-supervised learning into the training of the GNN.
- **COTREC** [38]: employs a self-supervised co-training approach. GCN encoders produce two views of a session on an item and session level for the contrastive learning task.
- **MGIR** [8]: models incompatible relations in a graph in addition to sequential and global co-occurrence.

- **DGNN** [18]: employs a dual graph neural network to model explicit and implicit depenedcies among items.

- **Atten-Mixer** [48]: drops redundant propagation modules and focuses on the read-out module to achieve multi-level reasoning over item transitions.

Implementation Details

Along the lines of previous works [8, 34, 38, 39], the embedding size is set to 100 and the parameters are initialized with a Gaussian distribution. For optimization, we use Adam with a learning rate of 0.001 and a batch size of 100. The L_2 regularization is set to 10^{-5} for all four datasets. Additionally, we apply a learning rate decay strategy, where the learning rate is decreased by a factor of 10 every 3 epochs. The maximum session length is set to 50 for all four datasets. The weighted normalization hyper-parameter w_z, the weight of the self-supervised loss β, and the number of samples k are searched in the ranges of $\{10, 11, 12, ..., 20\}$, $\{0.001, 0.005, ..., 0.5\}$ and $\{1, 2, 4, 8, 16, 32\}$, respectively. Since we use the same evaluation setup and datasets as the baseline methods, we adopt their best parameter setup and directly report their results if available. Our implementation is based on PyTorch 1.10.2 and Python 3.8.12. All experiments are performed on a workstation with an AMD Ryzen 2950X, a GeForce RTX 2070, and 256 GB main memory. We publish the code and the pre-processed datasets on GitHub[7].

6.5.2. Overall Performance (RQ1)

To demonstrate the recommendation performance of our proposed method, we compare SPARE with several other state-of-the-art and baseline SBR methods (see Baseline Methods). The overall performance on the four datasets is shown in Table 6.2. From this table, we can draw distinct conclusions which we will elaborate in the following.

Conventional methods like FPMC are outperformed by RNN-based methods (e. g., GRU4Rec, NARM, STAMP), which indicates the importance of modeling the sequential information of sessions. NARM and STAMP additionally incorporate an attention mechanism to learn item importance and show a large performance improvement compared to GRU4Rec. Since GRU4Rec only considers sequential behavior, it is not able to capture shifts in user preference.

Graph-based models easily outperform the aforementioned RNN-based methods and display the advantages of using graphs to model sessions. GCE-GNN and MGIR include inter- and intra-session information and are able to achieve a substantial performance boost compared to SR-GNN, demonstrating the importance of capturing different levels of information. S^2-DHCN and COTREC both have a two-branch architecture to

[7] https://github.com/dbis-uibk/SPARE

Table 6.2.: Model performance on all four datasets for baselines, state-of-the-art models (SotA), and our proposed SPARE approach. All improvements of SPARE compared to the second best performing model are significant (paired t-test, $p < .01$). The best results are in boldface and the second-best results are underlined.

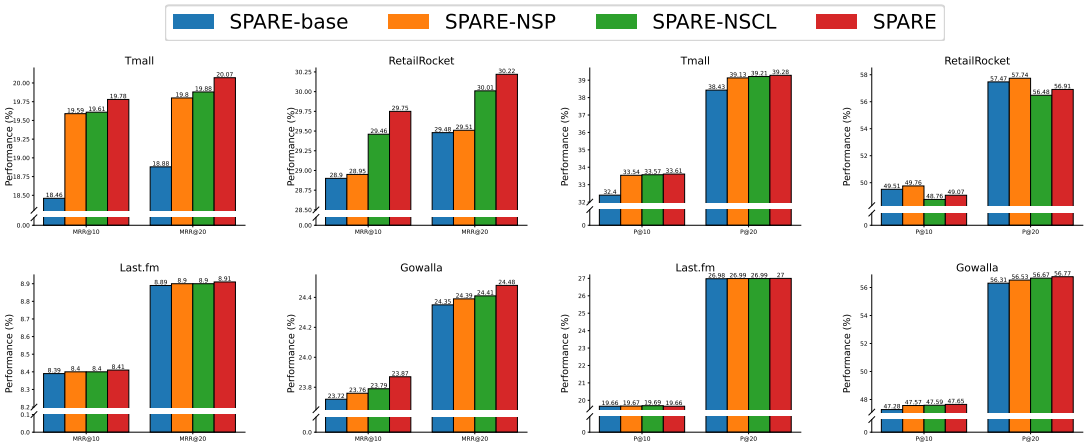
	Method	Tmall				RetailRocket				Last.fm				Gowalla			
		P		MRR		P		MRR		P		MRR		P		MRR	
		@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20
Baselines	Item-KNN	6.65	9.15	3.11	3.31	22.48	24.00	10.43	10.70	9.77	14.84	4.48	4.85	25.08	38.60	14.37	16.66
	FPMC	13.10	16.06	7.12	7.32	25.99	32.37	13.38	13.82	11.67	17.68	4.58	4.99	20.47	29.91	9.88	11.45
	GRU4Rec	16.59	20.39	9.05	9.31	38.35	44.01	23.27	23.67	12.86	17.90	5.29	5.39	31.56	41.91	17.85	18.29
	NARM	19.17	23.30	10.42	10.70	42.07	50.22	24.88	24.59	15.03	21.83	6.71	7.59	40.53	50.11	22.94	23.89
	STAMP	22.63	26.47	13.12	13.36	42.95	50.96	24.61	25.17	15.65	22.01	7.50	7.98	40.99	50.15	23.10	24.03
	SR-GNN	23.41	27.57	13.45	13.72	43.21	50.32	26.07	26.57	16.90	22.33	7.85	8.23	41.89	50.29	23.78	24.31
	FGNN	20.67	25.24	10.07	10.39	41.78	50.20	24.59	25.89	15.90	22.20	7.28	8.02	42.09	50.11	22.91	24.11
SotA	GCE-GNN	28.01	33.42	15.08	15.42	47.90	55.59	28.04	28.58	18.28	24.39	8.32	8.63	45.90	54.48	24.29	24.89
	S^2-DHCN	26.22	31.42	14.60	15.05	46.15	53.66	26.85	27.30	15.37	22.06	6.95	7.57	45.11	53.34	23.29	23.88
	COTREC	30.62	36.35	17.65	18.04	48.61	56.17	29.46	29.97	16.89	23.34	7.81	8.24	45.15	53.76	23.45	24.02
	MGIR	30.65	36.41	17.06	17.42	48.87	56.62	29.35	29.84	17.99	24.72	8.37	8.82	45.39	53.87	23.70	24.29
	DGNN	18.96	23.05	10.38	10.65	43.08	50.26	24.76	25.26	15.83	21.71	7.83	8.23	42.79	50.70	23.83	24.38
	Atten-Mixer	31.79	37.43	18.35	18.75	48.63	56.66	27.95	28.51	16.79	23.01	8.23	8.66	45.60	53.92	**26.35**	**26.93**
	SPARE	**33.61**	**39.28**	**19.78**	**20.07**	**49.07**	**56.91**	**29.75**	**30.22**	**19.66**	**27.00**	**8.41**	**8.91**	**47.65**	**56.77**	23.87	24.48
	Improv. (%)	5.72	4.94	7.79	7.04	0.41	0.44	0.98	0.83	7.55	9.22	0.48	0.91	3.81	4.20	-	-
	p-value	$1e^{-9}$	$7e^{-11}$	$7e^{-10}$	$1e^{-10}$	$9e^{-3}$	$2e^{-3}$	$6e^{-3}$	$6e^{-3}$	$4e^{-10}$	$4e^{-10}$	$3e^{-3}$	$2e^{-4}$	$1e^{-7}$	$1e^{-7}$	-	-

make use of a contrastive learning framework and are easily competitive. Specifically, COTREC and MGIR show superior performance to most of the graph-based models indicating the advantage of using self-supervised learning and global item graphs.

Our proposed method SPARE significantly surpasses all current state-of-the-art baseline methods on the first three datasets on all provided metrics. Particularly, our model improves the performance significantly by 5.72% on Precision@10 and 7.79% on MRR@10 for the *Tmall* dataset, showing the importance of dropping unreliable relations from e-commerce data. For *Gowalla* our approach reaches best performance for Precision and third-best for MRR compared to all models, which potentially shows that the particular task of point-of-interest recommendation inherits different characteristics than product or music recommendation. Especially, since Atten-Mixer, a non-graph-based model, achieves the best MRR overall on this dataset. Additionally, we observe that COTREC and MGIR have competitive performance on the *RetailRocket* and *Last.fm* datasets in terms of MRR. However, both of these methods introduce a complex architecture and have a higher running time compared to SPARE, limiting their practical applicability. We provide more details about the efficiency and running time of all current state-of-the-art models in Section 6.5.6.

6.5.3. Ablation Study (RQ2)

To investigate the impact of each component in our approach, including the shortest-path aware item graph (Section 6.4.1) and the supervised contrastive learning (Section 6.4.3), we present different variants of SPARE in this section: **SPARE-base**, **SPARE-NSP**,



(a) Mean Reciprocal Rank. (b) Precision.

Figure 6.3.: Ablation study of components in SPARE.

and **SPARE-NSCL**. In **SPARE-base** we omit the shortest-path search as well as the supervised contrastive learning. For **SPARE-NSP** only the shortest-path search on the global item graph is removed and in **SPARE-NSCL** only the supervised contrastive learning component is discarded. These models will be evaluated against the full SPARE model on all datasets. As previous studies [38, 39] already have shown that the reversed position embeddings and the soft-attention mechanism are important components, we discard these variants for our study.

In Figure 6.3a and Figure 6.3b we display the performances of all models in terms of Precision and MRR, both with cutoffs set to 10 and 20. It can be observed that each of our introduced components consistently contributes to the performance of the model. On the *Tmall*, *Last.fm* and *Gowalla* datasets both, the shortest-path search and the supervised contrastive learning, are able to improve the performance significantly if applied separately. Still, the integration of both components leads to the best-performing models on all metrics, showing that supervised contrastive learning complements the sparse, shortest-path graph representation learning. The evaluation on the *RetailRocket* dataset shows a slightly different result. Surprisingly SPARE-base is able to outperform SPARE in terms of Precision by a slight margin but heavily lacks in MRR performance. We ascribe this to the edge sparsification due to the shortest-path search which removes noisy items and increases the ranking of important ones, which has a positive impact on the MRR score. Also, performances on *Last.fm* seem to be not strongly affected by the different components. We speculate that this effect stems from the special characteristics of music datasets, which include longer user sessions as can be seen in Table 6.1. A lower intra-session sparsity reduces the impact of shortcut connections and contrastive learning techniques but increases the importance of item-item relation modeling. This effect will be investigated in Section 6.5.6.

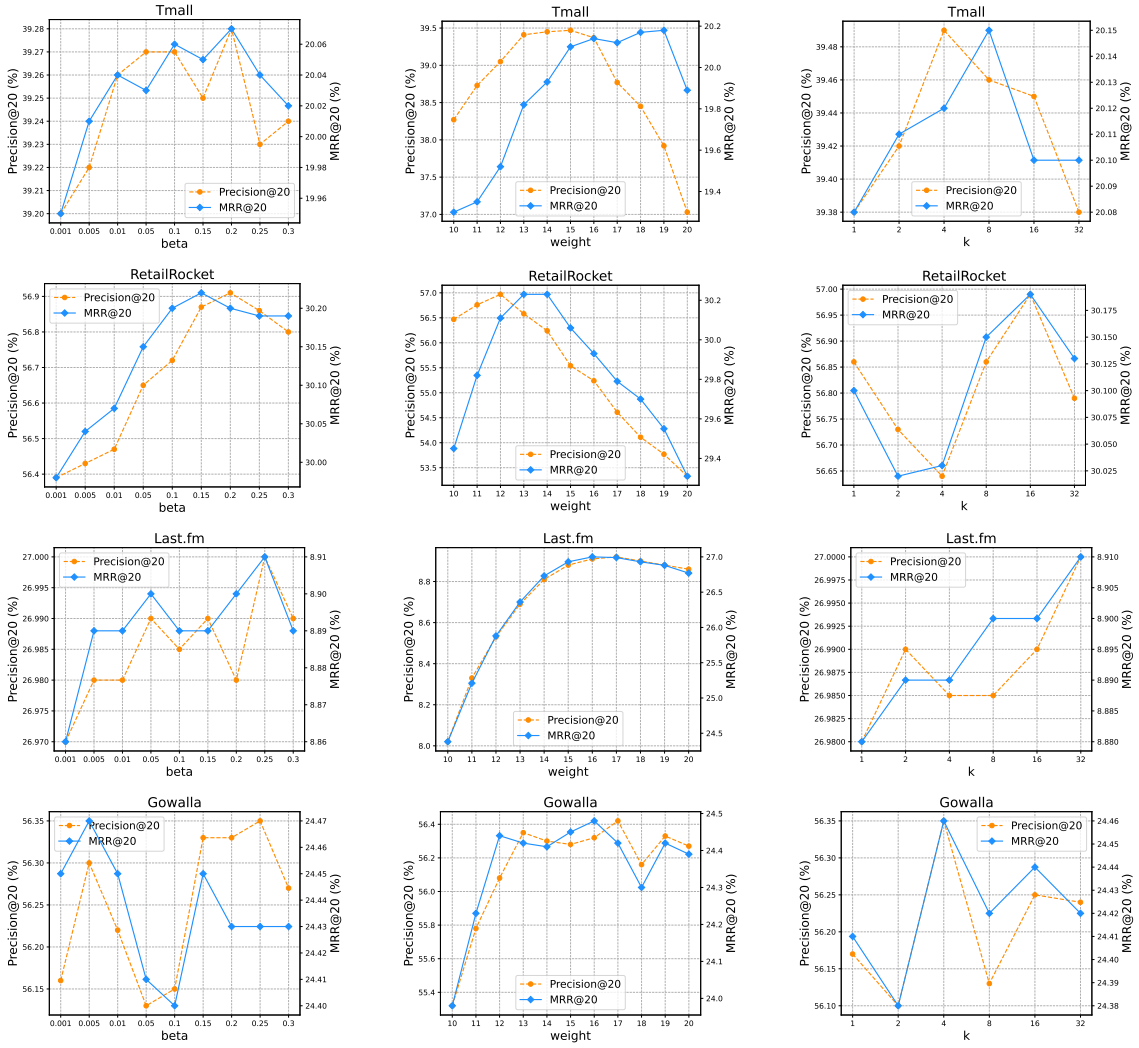
6.5.4. Impact of Hyper-Parameters (RQ3)

Furthermore, we investigate the impact of the three key hyper-parameters β (weight of the self-supervised loss), w_z (weighted L_2 normalization), and k (number of samples in SCL). The weight parameter β controls the magnitude of the self-supervised learning task and achieves the best performance if set to 0.2 and 0.15 for *Tmall* and *RetailRocket* as shown in Figure 6.4a. Since we optimize for MRR β is set to 0.25 and 0.05 for *Last.fm* and *Gowalla*, correspondingly. Additionally, we explore the influence of w_z, where setting it to 1 is equivalent to employing cosine similarity and delivers the poorest results. As w_z increases we observe a gradual improvement on all datasets until it oversaturates which can be seen in Figure 6.4b. This demonstrates the importance of this scaling factor to stabilize the training since target items with higher $\mathrm{L}_2 Norm$ are more prone to be predicted. In Figure 6.4c different settings for k corresponding to the number of positive and negative samples used in the supervised contrastive loss are displayed. It can be observed that on the *Tmall* and *Gowalla* datasets, a smaller number is sufficient, whereas the *RetailRocket* and *Last.fm* datasets benefit from a higher number of samples.

Furthermore, we investigate the impact of the hyper-parameter μ (cost limit for shortest paths) on the sparsity of the global item graph and the model performance. The sparsity value per cost limit (or rather, the increase of sparsity) is defined as follows:

$$Sparsity = 1 - \frac{\#\text{ edges}}{\#\text{ original edges}}, \tag{6.17}$$

which defines the ratio of increase or decrease of sparsity compared to the original global item graph and allows us to directly investigate the relationship between higher sparsity (more reduced noise) and prediction performance. In Figure 6.5 the sparsity and the MRR@20 score per dataset are displayed. These analyses show that higher sparsity of the global item graph and therefore possibly dropping unreliable relations for *Tmall* and *RetailRocket* has a considerable impact on the performance. It is worth noting, that for *Tmall* and *RetailRocket* the maximum edge cost in the original global item graph is 197 and 331 correspondingly. For *Last.fm* and *Gowalla*, we observe a different behavior: Instead of filtering out non-frequent relations by setting μ below the maximum edge cost, we reach better performance by using a limit that is the same as the maximum edge cost (1526 and 153) and therefore introducing a slightly denser global item graph through the addition of shortest-path shortcut connections. In the case of *Last.fm* we ascribe this to the inherent characteristics of music datasets compared to other domains to be more prone to the popularity bias of songs [16] and benefit more from dense user data for personalized recommendations [9]. A similar explanation can be provided for the *Gowalla* dataset, which can also be affected by over-popular points of interest. This particularly shows the data-driven versatility of our proposed method, being able to adapt to different data sparsity conditions. This special behavior of SPARE on *Last.fm* is analyzed in more detail in Section 6.5.6, where we show that on this dataset our approach also benefits from a higher number of layers in the GNN.



(a) Weight parameter β for self-supervised loss.

(b) Weight parameter w_z of L_2 normalization.

(c) Number of samples parameter k for computing the contrastive loss.

Figure 6.4.: Impact of hyper-parameters in SPARE.

6.5.5. Impact of Supervised Contrastive Learning (RQ4)

In Section 6.4.3 we introduced BLEU as a measure for session similarity. To justify this design choice and display the impact of the supervised contrastive learning component in our model we compare different session similarity measures. Nearest-neighbor-based methods usually rely on session similarity measures to filter out relevant sessions for the computation of potential next-item candidates [5, 12, 20]. Following their intuition of defining sessions as a set of items we include the following set-based similarity

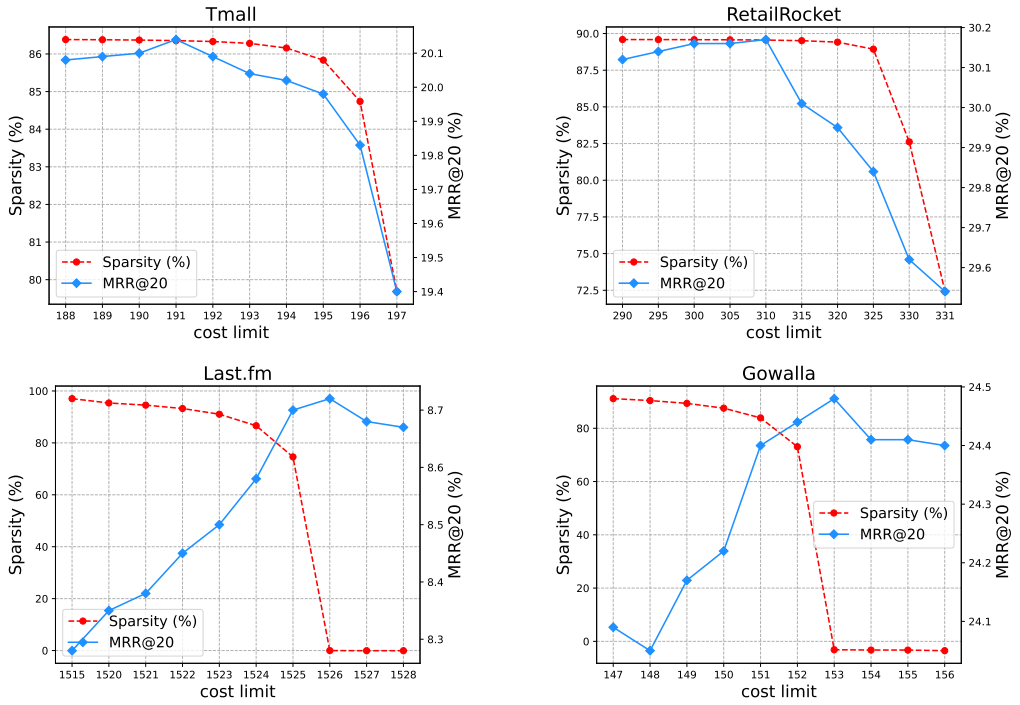


Figure 6.5.: Different cost limits μ for shortest-path search affect sparsity of the original global item graph and have an impact on the corresponding MRR@20 performance. Sparsity is defined as the fraction (given in %) of additional pruned edges compared to the number of edges in the original item graph.

measures: Cosine-similarity (for sets) and Jaccard-index. Additionally, we also explore the Damerau-Levenshtein distance, which is usually used to measure the edit distance between two sequences, in the comparison. As shown in Table 6.3 most of the different session similarity measures are able to improve the performance of the base model without supervised contrastive learning. Interestingly, the set-based measures perform better than the more sequence-oriented Damerau-Levenshtein distance. Nevertheless, BLEU with its n-gram overlap-dependent measurement outperforms on average all other session similarity measures and shows the importance of considering the sequential nature of sessions. Importantly, the positive impact on both performance metrics through supervised contrastive learning can be seen. On the *Tmall* and *RetailRocket* datasets, incorporating the supervised contrastive learning loss leads to an increase in performance of 0.17% and 0.76% in Precision@20 and 0.95% and 0.69% in MRR@20, correspondingly. Other similarity measures like Jaccard-index or Cosine-similarity seemingly can improve the performance on different datasets, but introduce a trade-off between Precision and MRR, whereas BLEU is the only measure to deliver consistent improvements, across all datasets and metrics. We also compare our SCL approach with an self-supervised variant (**SPARE-SSL**) which does not use any label information to extract positive and negative

Table 6.3.: Comparison of different distance measures for session similarity computation.

Similarity	Tmall		RetailRocket		Last.fm		Gowalla	
	P@20	MRR@20	P@20	MRR@20	P@20	MRR@20	P@20	MRR@20
SPARE-NSCL	39.21	19.88	56.48	30.01	26.97	8.90	56.67	24.41
SPARE-SSL	39.14	20.04	56.52	30.12	26.86	8.90	56.72	24.39
Cosine	39.12	19.96	56.69	30.05	**27.00**	8.91	56.29	24.43
Jaccard	39.23	20.02	56.79	30.09	26.96	**8.93**	56.52	**24.52**
Levenshtein	39.08	19.95	56.74	30.03	26.99	8.91	56.39	24.41
BLEU	**39.28**	**20.07**	**56.91**	**30.22**	**27.00**	8.91	**56.77**	24.48

samples. We use random masking of sessions for positive samples and sample random sessions from the batch for negative samples. Although the self-supervised variant is competitive with some of the different distance measures, it is clearly outperformed by our SCL using BLEU similarity. This underlines the importance of using label information to sample more informative positive and hard negative samples for the contrastive loss.

6.5.6. Impact of Number of Layers and Running Times (RQ5)

We hypothesize that SPARE through its shortest-path shortcut connections inherently introduces a large receptive field per node. Consequently, SPARE does not have to rely on multiple layers to aggregate node information from neighbors multiple hops away. To confirm this intuition we compare SPARE-NSCL (our model without the supervised contrastive learning component) and COTREC (the model showing the second-best overall performance) with a different number of layer settings, since they use a similar graph convolutional operation.

To combine learned node embeddings over multiple layers, we follow the strategy of COTREC, where item embeddings are averaged over L layers to get the final embeddings:

$$X^{(L)} = \frac{1}{L+1} \sum_{l=0}^{L} X^l. \tag{6.18}$$

Figure 6.6 exhibits the results of these experiments on all four datasets. We can observe that COTREC heavily relies on learning the item representations in the graph by using information from n-hop neighbors and constantly reaches its best performance in the 3-layer setting, but suffers from oversmoothing with a higher number of layers. In contrast, our proposed method SPARE has stable performance across all settings of layers, indicating that multi-hop connections are effectively captured by shortest-path shortcut connections. Notably, on the *Last.fm* dataset our model is able to constantly improve its performance with a higher number of layers and is not affected by the over-smoothing

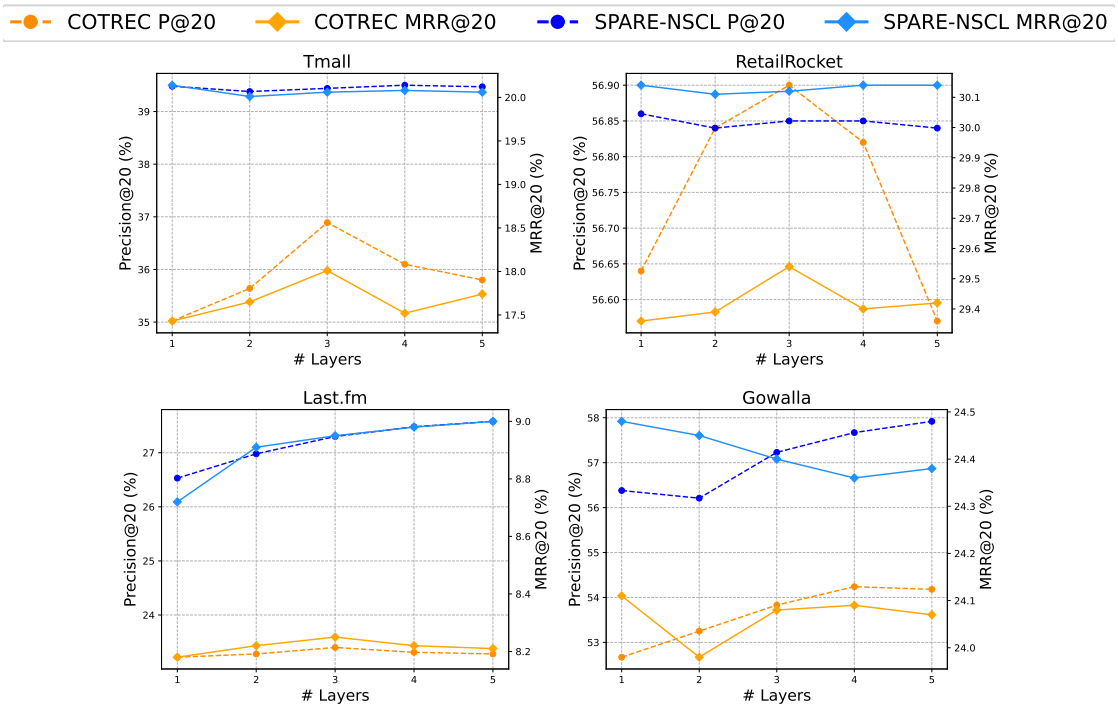


Figure 6.6.: Comparison of Precision@20 and MRR@20 of COTREC [38] versus SPARE-NSCL dependent on different number of layers.

issue. We assume this behavior is due to the inherent popularity bias of the dataset so that a larger receptive field per node stabilizes the training which is also indicated in Figure 6.5, where a lower data sparsity seems beneficial.

Our model introduces a simple, yet effective architecture and mostly has to rely on only a single GNN layer (except for *Last.fm*) to compute the global item embeddings. To demonstrate the practicability of our approach, we compare the running times as a proxy for efficiency for SPARE and state-of-the-art graph-based methods for SBR (based on Table 6.2) on all four datasets. Similar to previous approaches [8] we define the graph construction as external pre-processing step and do not include this step in the running time measurement. Although the positive and negative session sampling is a CPU-bound operation and can easily be parallelized, we include them in the measurement for a fair comparison. The running times per model are averaged over 5 epochs.

As shown in Table 6.4 our approach has the fastest running time on the *Tmall* as well as the *RetailRocket* and the *Gowalla* datasets. To be more specific, SPARE is able to reach a speed-up factor of 1.84× compared to the fastest graph-based method (GCE-GNN) on *RetailRocket*. If compared to the second-best performing model in terms of P@20

Table 6.4.: Comparison of training running times per epoch per graph-based SotA method (in seconds).

Method	Tmall	RetailRocket	Last.fm	Gowalla
GCE-GNN	116	1,154	**832**	182
S^2-DHCN	664	1,313	14,453	1221
COTREC	1,170	1,233	5,220	1085
MGIR	448	1,344	2,408	242
SPARE	**105**	**624**	1,540	**171**

and MRR@20 on *Last.fm* (MGIR), as shown in Table 6.2, SPARE is faster by 1.56× in training. This clearly indicates that our approach learns global item representations more efficiently than every other state-of-the-art graph-based method.

6.5.7. Handling Different Session Lengths (RQ6)

In the dynamic and ever-evolving domain of session-based recommendations, the stability and adaptability of recommendation models are crucial, especially in real-world scenarios where sessions vary significantly in length [38]. To assess the robustness of SPARE in handling sessions of different lengths, we conduct a comparative study using a range of well-established models: GRU4Rec, SR-GNN, GCE-GNN, and COTREC. For this study, we follow previous works [19, 38] where each dataset gets divided into two distinct session length groups: *Short* and *Long*. The pivot value to differ between *Short* and *Long* sessions is chosen to be the closest integer to the average length of all sessions in each dataset. For simplicity the *Short* group for the datasets *Tmall*, *RetailRocket* and *Gowalla* encompasses sessions with lengths of five interactions or less, while the *Long* group includes sessions exceeding five interactions. For the *Last.fm* dataset we chose the pivot value to be 12, since it has a much higher average session length (cf. Table 6.1).

In Figure 6.7 it can be observed that all models generally perform better on *Long* sessions compared to *Short* sessions, except for the *RetailRocket* dataset. This pattern demonstrates the capability of all models to capture more complex user interests as session length increases, despite evolving interests and potential added noise in longer sessions. All graph-based models show commendable performance in both session groups compared to GRU4Rec, indicating their robustness in handling varied session lengths. GCE-GNN and COTREC, while effective, exhibit a slight drop in performance in *Short* sessions (e. g., *Tmall* and *Gowalla*), hinting at potential challenges in managing short-term user interests. SPARE stands out for its exceptional adaptability, consistently delivering strong results across both session groups. Especially on *Last.fm* SPARE can boost the recommendation performance for short sessions by 8.73% compared to the next best-performing model GCE-GNN.

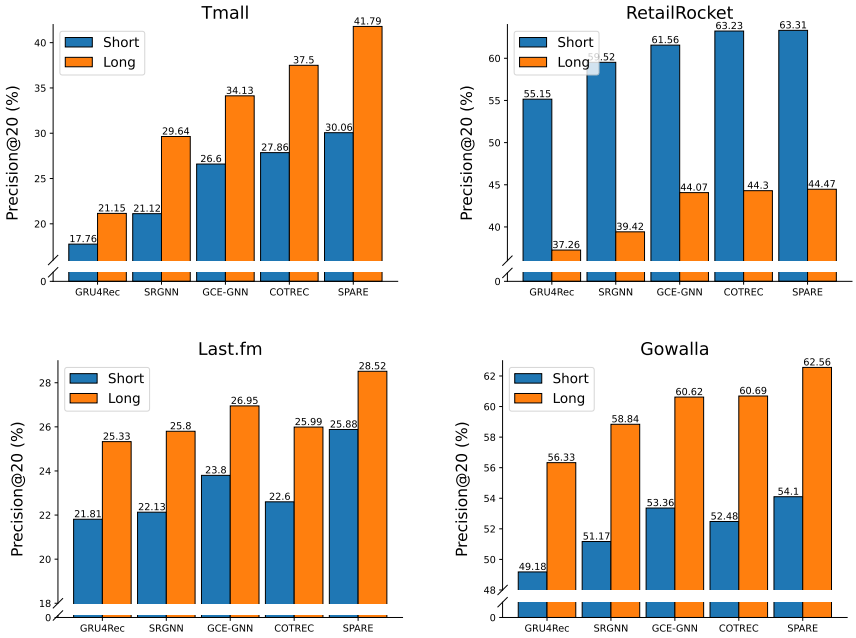


Figure 6.7.: P@20 results on Short and Long sessions.

Table 6.5.: Comparative performance and efficiency metrics of S^2-DHCN and COTREC models with and without SPARE's graph-building strategy across four datasets. Improvement percentages are calculated relative to the baseline models without SPARE's graph-building strategy (w/o) versus with the strategy (w/).

Method		Tmall					Last.fm					Gowalla				
		P		MRR			P		MRR			P		MRR		
		@10	@20	@10	@20	Time (s)	@10	@20	@10	@20	Time (s)	@10	@20	@10	@20	Time (s)
S^2-DHCN	w/o	26.22	31.42	14.60	15.05	*664*	15.37	22.06	6.95	7.57	*14,453*	45.11	53.34	23.29	23.88	*1,221*
	w	**27.15**	**32.79**	**15.23**	**15.75**	***576***	**17.24**	**23.89**	**7.81**	**8.26**	*4,714*	**46.16**	**54.94**	**23.96**	**24.56**	***1,004***
Improv. (%)		3.54	3.09	4.36	4.62	13.25	12.16	8.29	12.37	9.11	67.38	2.32	2.99	2.87	2.84	17.77
COTREC	w/o	30.62	36.35	17.65	18.04	*1,170*	16.89	23.34	**7.81**	8.24	*5,220*	**45.15**	**53.76**	23.45	24.02	*1,085*
	w	**31.11**	**37.10**	**17.80**	**18.36**	***689***	**17.00**	**23.47**	7.15	**8.25**	***1,170***	44.44	52.64	**23.53**	**24.09**	***1,033***
Improv. (%)		1.61	2.05	0.89	1.77	41.11	0.65	0.55	-	0.12	77.58	-	-	2.72	0.29	4.79

6.5.8. SPARE Enhancement Study (RQ7)

This enhancement study focuses on the performance improvement of baseline recommendation models, specifically S^2-DHCN and COTREC, when augmented with the graph-building strategy derived from the SPARE model. The study aims to establish whether the integration of SPARE's strategy can lead to enhanced recommendation performance and training efficiency. Both S^2-DHCN and COTREC models have a similar graph processing pipeline as SPARE and are adapted to incorporate SPARE's graph-building

strategy including the shortest-path search. We hypothesize that our approach (relying only on a single GNN layer) can outperform the baseline approaches (using three GNN layers) in terms of performance and training efficiency.

The results, as presented in Table 6.5, indicate a substantial improvement across three different datasets: *Tmall*, *Last.fm*, and *Gowalla*. The *RetailRocket* dataset (another e-commerce dataset similar to *Tmall*) is neglected for this study due to space reasons. For S^2-DHCN, we observe notable performance gains in Precision and MRR. S^2-DHCN sees an average improvement of 10.22% in Precision and 10.74% in MRR on the Last.fm dataset, with *Tmall* and *Gowalla* also demonstrating notable gains. Notably, these enhancements come with a substantial decrease in training time (up to 67.38%), underscoring the efficiency of the SPARE-inspired strategy.

Similarly, the performance of COTREC, when enhanced with SPARE's strategy, shows improvement on most of the datasets in Precision and MRR. COTREC shows a promising enhancement, with a 2.05% improvement in Precision@20 and a 1.77% increase in MRR@20 on *Tmall*. Similar positive trends are observed with *Last.fm* and *Gowalla*, although there is room for further advancement in achieving competitive performances across all individual metrics. A possible reason for this effect could be the graph augmentations in COTREC, which were probably not intended to be used with custom graph structures. The training times are also reduced significantly, suggesting that the graph-building strategy not only enhances recommendation quality but also optimizes computational efficiency.

The study confirms that the integration of SPARE's graph-building strategy into baseline models like S^2-DHCN and COTREC results in a significant performance enhancement. This improvement is consistent across various metrics and datasets, emphasizing the robustness of the strategy. For COTREC, we are able to improve on 9 out of 12 metrics, for S^2-DHCN, it's even 12 out of 12. Furthermore, the reduction in training times highlights the strategy's added benefit of efficiency, making it a possible candidate strategy for future graph-based SBR.

6.5.9. Graph Structure Case Study (RQ8)

To demonstrate the nuanced capability of our SPARE model in delivering personalized music recommendations (e.g. on *Last.fm*), we provide a qualitative view of the graph structure and analyze a specific case involving a random user session identified by the session s_{171275} (as depicted in Figure 6.8). This user has a multifaceted listening history that includes a wide range of genres, from rock over hardcore to classical. For the user in question, the session's listening sequence begins with U2 (a renowned rock band), transitioning through various genres including hardcore (Evil Activities), reggae (Damian

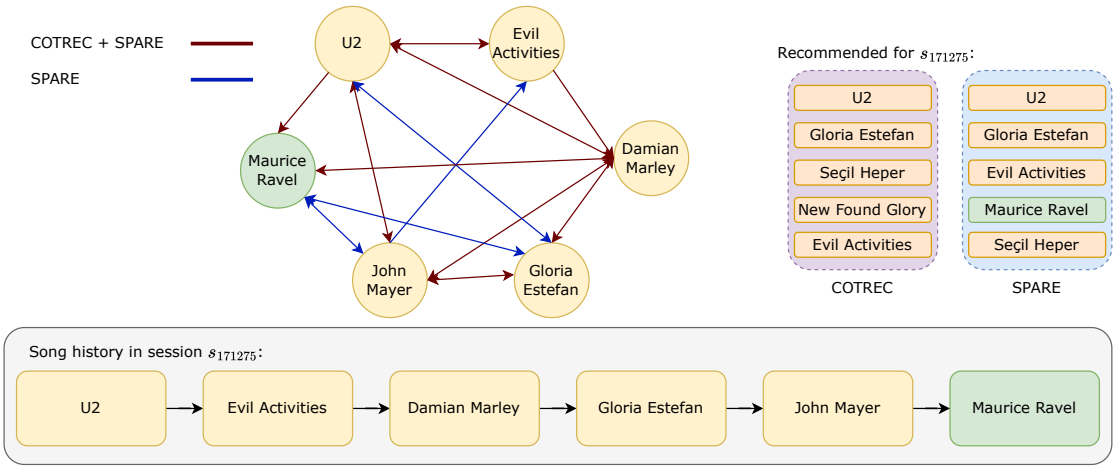


Figure 6.8.: A case study of one user session from Last.fm data for music artist recommendation.

Marley), Latin pop (Gloria Estefan), contemporary rock (John Mayer), and culminating with Maurice Ravel, an iconic classical composer. This eclectic mix indicates a user with diverse and complex music preferences.

As shown in Figure 6.8, COTREC and SPARE generated graphs differ in their interconnections. COTREC mainly models the sequential dependencies from the sessions, whereas SPARE can reach higher connectivity of each node due to its shortcut connections introduced by the shortest-path search (blue edges). For instance, it connects Maurice Ravel with artists from different genres, indicating a recognition of the user's appreciation for both classical compositions and their intricate musicality, which may also be present in rock and pop music. This contrasts with COTREC's recommendations, which, while varied, lack the personalized depth SPARE provides. While COTREC suggests Segil Heper and New Found Glory, which may cater to a more general audience, SPARE identifies connections with artists like Gloria Estefan and Maurice Ravel, aligning with the user's demonstrated interest in both Latin rhythms and classical music.

6.6. Conclusion

Session-based recommendation exhibits many challenges including sparse session data, anonymous users, and current preference shifts. In this paper, we propose a novel session-based recommendation model that relies on a shortest-path search to filter out unreliable relations and to introduce shortcut connections to items multiple hops away for a dense graph representation. Moreover, we present a novel supervised contrastive learning method based on data-driven positive and negative item samples for SBR. To find hard negative samples we propose to use the BLEU metric to find similar sessions to

the reference sessions. An extensive experimental evaluation comparing with different state-of-the-art models shows the effectiveness of our approach and its superiority over other baseline models.

In future work, we plan to use the denoised global item graphs to extract explainable recommendations. Furthermore, we aim to investigate the impact of supervised contrastive learning in combination with weighted L_2 normalization on improving popularity bias. Potentially, these techniques can be applied to a various number of other methods in an extension-like fashion, some of which we even have shown in this paper.

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499. Morgan Kaufmann Publishers Inc., 1994. ISBN: 1558601538.

[2] T. Chen and R. C. Wong. Handling information loss of graph neural networks for session-based recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1172–1180. ACM, 2020.

[3] G. de Souza Pereira Moreira, S. Rabhi, J. M. Lee, R. Ak, and E. Oldridge. Transformers4rec: bridging the gap between NLP and sequential / session-based recommendation. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems*, pages 143–153. ACM, 2021.

[4] M. L. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3):596–615, 1987.

[5] D. Garg, P. Gupta, P. Malhotra, L. Vig, and G. Shroff. Sequence and time aware neighborhood for session-based recommendations: STAN. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019*, pages 1069–1072. ACM, 2019.

[6] L. Guo, H. Yin, Q. Wang, T. Chen, A. Zhou, and N. Q. V. Hung. Streaming session-based recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019*, pages 1569–1577. ACM, 2019.

[7] P. Gupta, D. Garg, P. Malhotra, L. Vig, and G. M. Shroff. Niser: normalized item and session representations with graph neural networks. *arXiv preprint arXiv:1909.04276*, 2019.

[8] Q. Han, C. Zhang, R. Chen, R. Lai, H. Song, and L. Li. Multi-faceted global item relation learning for session-based recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1705–1715. ACM, 2022.

[9] C. Hansen, C. Hansen, L. Maystre, R. Mehrotra, B. Brost, F. Tomasi, and M. Lalmas. Contextual and sequential user embeddings for large-scale music recommendation. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems*, pages 53–62. ACM, 2020.

[10] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, Conference Track Proceedings*, 2016.

[11] C. Hsu and C. Li. Retagnn: relational temporal attentive graph neural networks for holistic sequential recommendation. In *WWW '21: The Web Conference 2021*, pages 2968–2979. ACM, 2021.

[12] D. Jannach and M. Ludewig. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017*, pages 306–310. ACM, 2017.

[13] W. Kang and J. J. McAuley. Self-attentive sequential recommendation. In *IEEE International Conference on Data Mining, ICDM 2018*, pages 197–206. IEEE Computer Society, 2018.

[14] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan. Supervised contrastive learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.

[15] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.

[16] D. Kowald, M. Schedl, and E. Lex. The unfairness of popularity bias in music recommendation: A reproducibility study. In *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020*, volume 12036 of *Lecture Notes in Computer Science*, pages 35–42. Springer, 2020.

[17] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017*, pages 1419–1428. ACM, 2017.

[18] Z. Li, X. Wang, C. Yang, L. Yao, J. J. McAuley, and G. Xu. Exploiting explicit and implicit item relationships for session-based recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM 2023*, pages 553–561. ACM, 2023.

[19] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018*, pages 1831–1839. ACM, 2018.

[20] M. Ludewig, I. Kamehkhosh, N. Landia, and D. Jannach. Effective nearest-neighbor music recommendations. In *Proceedings of the ACM Recommender Systems Challenge, RecSys Challenge 2018*, 3:1–3:6. ACM, 2018.

[21] C. Ma, P. Kang, and X. Liu. Hierarchical gating networks for sequential recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019*, pages 825–833. ACM, 2019.

[22] C. Ma, L. Ma, Y. Zhang, J. Sun, X. Liu, and M. Coates. Memory augmented graph neural networks for sequential recommendation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pages 5045–5052. AAAI Press, 2020.

[23] K. Papineni, S. Roukos, T. Ward, and W. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318. ACL, 2002.

[24] A. Peintner, A. R. Mohammadi, and E. Zangerle. SPARE: shortest path global item relations for efficient session-based recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, pages 58–69. ACM, 2023.

[25] R. Qiu, J. Li, Z. Huang, and H. Yin. Rethinking the item order in session-based recommendation with graph neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019*, pages 579–588. ACM, 2019.

[26] M. Quadrana, P. Cremonesi, and D. Jannach. Sequence-aware recommender systems. *ACM Computing Surveys (CSUR)*, 51(4):1–36, 2018.

[27] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010*, pages 811–820. ACM, 2010.

[28] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10*, pages 285–295. ACM, 2001.

[29] Q. Tan, J. Zhang, N. Liu, X. Huang, H. Yang, J. Zhou, and X. Hu. Dynamic memory based attention network for sequential recommendation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, pages 4384–4392. AAAI Press, 2021.

[30] J. Tang and K. Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018*, pages 565–573. ACM, 2018.

[31] T. X. Tuan and T. M. Phuong. 3d convolutional networks for session-based recommendation with content features. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017*, pages 138–146. ACM, 2017.

[32] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

[33] M. Wang, P. Ren, L. Mei, Z. Chen, J. Ma, and M. de Rijke. A collaborative session-based recommendation approach with parallel memory modules. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019*, pages 345–354. ACM, 2019.

[34] Z. Wang, W. Wei, G. Cong, X. Li, X. Mao, and M. Qiu. Global context enhanced graph neural networks for session-based recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, pages 169–178. ACM, 2020.

[35] F. Wu, A. H. S. Jr., T. Zhang, C. Fifty, T. Yu, and K. Q. Weinberger. Simplifying graph convolutional networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97 of *Proceedings of Machine Learning Research*, pages 6861–6871. PMLR, 2019.

[36] L. Wu, S. Li, C. Hsieh, and J. Sharpnack. SSE-PT: sequential recommendation via personalized transformer. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems*, pages 328–337. ACM, 2020.

[37] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan. Session-based recommendation with graph neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, volume 33 of number 01, pages 346–353. AAAI Press, July 2019.

[38] X. Xia, H. Yin, J. Yu, Y. Shao, and L. Cui. Self-supervised graph co-training for session-based recommendation. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*, pages 2180–2190. ACM, 2021.

[39] X. Xia, H. Yin, J. Yu, Q. Wang, L. Cui, and X. Zhang. Self-supervised hypergraph convolutional networks for session-based recommendation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, pages 4503–4511. AAAI Press, 2021.

[40] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, and X. Zhou. Graph contextualized self-attention network for session-based recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 3940–3946. ijcai.org, 2019.

[41] C. Xu, P. Zhao, Y. Liu, J. Xu, V. S. Sheng, Z. Cui, X. Zhou, and H. Xiong. Recurrent convolutional neural network for sequential recommendation. In *The World Wide Web Conference, WWW 2019*, pages 3398–3404. ACM, 2019.

[42] Y. Yang, X. Wang, M. Song, J. Yuan, and D. Tao. SPAGAN: shortest path graph attention network. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 4099–4105. ijcai.org, 2019.

[43] J. Yu, H. Yin, X. Xia, T. Chen, L. Cui, and Q. V. H. Nguyen. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In SIGIR '22, pages 1294–1303. ACM, 2022.

[44] L. Yu, C. Zhang, S. Liang, and X. Zhang. Multi-order attentive ranking model for sequential recommendation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 5709–5716. AAAI Press, 2019.

[45] F. Yuan, A. Karatzoglou, I. Arapakis, J. M. Jose, and X. He. A simple convolutional generative network for next item recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019*, pages 582–590. ACM, 2019.

[46] J. Yuan, Z. Song, M. Sun, X. Wang, and W. X. Zhao. Dual sparse attention network for session-based recommendation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, pages 4635–4643. AAAI Press, 2021.

[47] E. Zangerle and C. Bauer. Evaluating Recommender Systems: Survey and Framework. *ACM Computing Surveys*, 55(8):170:1–170:38, 2022.

[48] P. Zhang, J. Guo, C. Li, Y. Xie, J. Kim, Y. Zhang, X. Xie, H. Wang, and S. Kim. Efficiently leveraging multi-level user intent for session-based recommendation via atten-mixer network. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM 2023*, pages 168–176. ACM, 2023.

[49] T. Zhang, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, D. Wang, G. Liu, and X. Zhou. Feature-level deeper self-attention network for sequential recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 4320–4326. ijcai.org, 2019.

[50] Y. Zhang, Y. Liu, Y. Xu, H. Xiong, C. Lei, W. He, L. Cui, and C. Miao. Enhancing sequential recommendation with graph contrastive learning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022*, pages 2398–2405. ijcai.org, 2022.

[51] K. Zhou, H. Wang, W. X. Zhao, Y. Zhu, S. Wang, F. Zhang, Z. Wang, and J. Wen. S3-rec: self-supervised learning for sequential recommendation with mutual information maximization. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management*, pages 1893–1902. ACM, 2020.

7. Hypergraph-based Temporal Modelling of Repeated Intent for Sequential Recommendation

Publication

A. Peintner, A. R. Mohammadi, M. Müller, and E. Zangerle. Hypergraph-based temporal modelling of repeated intent for sequential recommendation. In *Proceedings of the ACM on Web Conference 2025, WWW 2025*, pages 3809–3818. ACM, 2025. DOI: 10.1145/3696410.3714896

Abstract

In sequential recommendation scenarios, user intent is a key driver of consumption behavior. However, consumption intents are usually latent and hence, difficult to leverage for recommender systems. Additionally, intents can be of repeated nature (e. g., yearly shopping for christmas gifts or buying a new phone), which has not been exploited by previous approaches. To navigate these impediments we propose the *HyperHawkes* model which models user sessions via hypergraphs and extracts user intents via soft clustering. We use Hawkes Processes to model the temporal dynamics of intents, namely repeated consumption patterns and long-term interests of users. For short-term interest adaption, which is more fine-grained than intent-level modeling, we use a multi-level attention mixture network and fuse long-term and short-term signals. We use the generalized expectation-maximization (EM) framework for training the model by alternating between intent representation learning and optimizing parameters of the long- and short-term modules. Extensive experiments on four real-world datasets from different domains show that HyperHawkes significantly outperforms existing state-of-the-art methods.

7.1. Introduction

Recommender systems have long become essential in filtering information effectively, for instance on video-sharing websites, e-commerce platforms, online bookstores, and social networks. With the abundance of online information, recommender systems have gained increasing importance by discovering and leveraging the underlying (latent) intents of users to cater to their preferences. In recent years, there has been a growing trend in modeling user sequential behaviors, which aims to capture short-term user interest and longer-term sequential patterns including popularity trends and interest drifts [42]. While traditional recommendation methods focus on static user preference modeling [16, 45], Sequential Recommendation (SR) models dynamically characterize user behaviors [18, 24], aiming to accurately predict users' interests in items based on their historical interactions and their corresponding points in time, allowing for more accurate and timely recommendations [8, 55].

The majority of previous works in SR order items by interaction timestamps and focus on sequential patterns to predict the next potential item. Early works adopt Markov chains to provide recommendations based on the L previous interactions via an L-order Markov chain [15, 46]. Also, Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) have been applied to model long- and short-term dependencies in a user interaction sequence [18, 67]. More recent methods rely on the self-attention mechanism and transformer-based models for capturing complex sequential dependencies for next-item recommendations [24, 49]. Another line of work explicitly focuses on modeling temporal dynamics in item sequences based on interaction timestamps [31, 66]. The availability of temporal information also enables models to learn about global events (e. g., Christmas) [56] and the periodicity of items [4, 55]. Previous works in the field model the temporal dynamics on an item level or rely on additional category and knowledge-graph information to represent user intent [19, 54]. However, these approaches

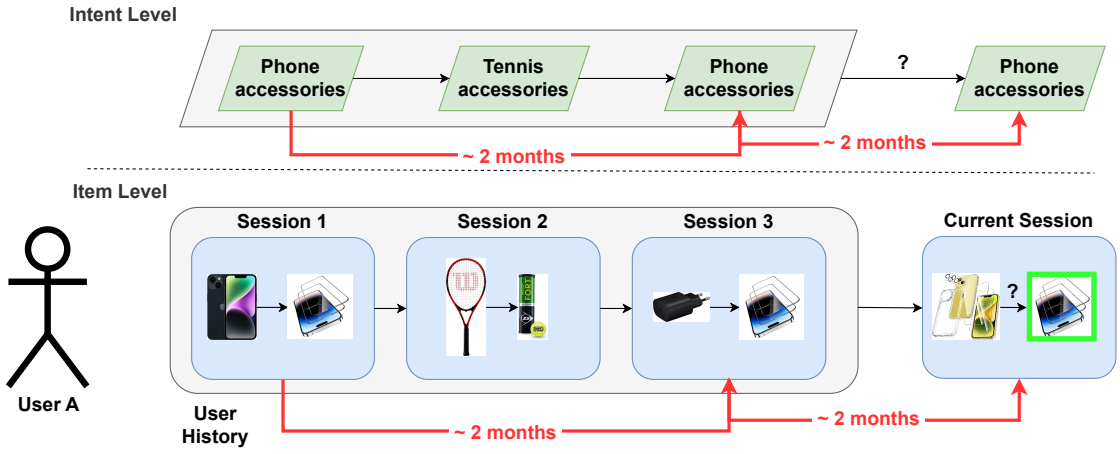


Figure 7.1.: A toy example of an e-commerce retailer scenario with repeated user intents.

come with several downsides: Learning temporal dynamics on the item level is often difficult due to data sparsity and ignores co-occurring item consumption patterns across all users. Also, valuable meta-information for learning user intents is not always available and mostly ignores personal user preferences like preferred brands, price restrictions, or re-consumption behavior.

To fill the aforementioned gaps, we propose to extract latent user intents from the user interaction sequences and model personalized temporal dynamics including repeat consumption on the user intent level. Consider the example in Figure 7.1. During each session, User A interacts with the system by e.g., viewing or purchasing items with different intents, and in this example, their interest is solely focused on the items relevant to their current intent. From the user's interaction history, it is apparent that the intent of consuming phone accessories is of repeated nature and is connected to the lifetime of a screen protector for the phone. Explicitly modeling this behavior increases the ability to recommend suitable phone accessories after a certain period (e. g., two months).

Repeat consumption occurs due to people's habits. For instance, we frequently purchase the same items, dine at the same restaurants, and listen to the same songs and artists often with a certain intent [1]. To empirically analyze the intent repeat consumption in the real world, we extract sets of frequently co-occurring item sets via the FP-Max algorithm [13]. For each active user (a user with at least 20 item interactions) we compute maximum frequent item sets (appearing twice or more in the user history) with a size larger than 1 to capture intent-level interactions. Then, we compute the maximum support of all repeated intents per user, where a support of 0.5 of an item set means this intent is apparent in 50% of the user's sessions. Figure 7.2 displays the distribution of intent repeat consumption with different maximum support values for four real-world benchmark datasets from different domains. Although there is a large portion of users with non-repeating intents, it is clear to see that intent repeat consumption is prevalent, and also constitutes a significant portion of interactions in certain domains.

To bridge this gap of modeling temporal dynamics of user intents we propose the *Hyper*graph-based *Hawkes* Processes (*HyperHawkes*) model for sequential recommendation. Our approach leverages hypergraphs and soft clustering to extract latent user intent representations from the user interaction data. Based on these user intent representations our temporal excitation module learns the dynamics of user intents and item consumption behavior based on Hawkes Processes [14], a temporal point process to model discrete events in a continuous-time regime. We propose a novel time decay function to represent the excitation strength between historical intent and item behaviors and their corresponding time intervals. To capture short-term interest changes on the item level, we additionally compute short-term interest scores based on an attention mixture network, which captures the influence of the last interacted items in the current session. These

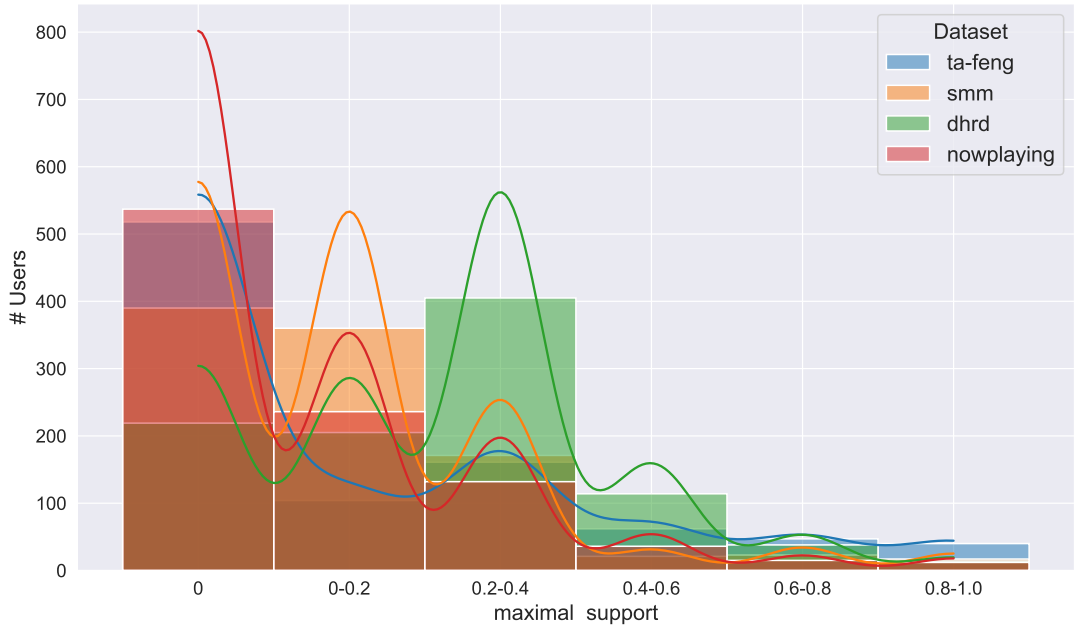


Figure 7.2.: Distribution of maximal support of intents (item sets with size >= 2) of active users per dataset. We randomly sampled 1000 users per dataset to ensure comparability between datasets.

steps ensure that our model effectively combines long-term and short-term user interest, and models both intent- and item-level temporal dynamics. We summarize our main technical contributions as follows:

- We propose a novel global item hypergraph construction strategy for learning intent-based item representations and employ soft clustering to extract latent user intents.
- We integrate Hawkes Processes (temporal point processes) to model long-term temporal dynamics on intent level; further, we fuse short-term interests for increased personalized recommendation performance.
- We conduct extensive experiments showing that our proposed model achieves significant performance improvements over a large number of state-of-the-art competitors on four datasets from different domains.[1]

[1] Code: https://github.com/dbis-uibk/HyperHawkes

7.2. Related Work

In this section, we review related work, which includes sequential recommendation, user intent, and temporal information learning.

7.2.1. Sequential Recommendation

Sequential recommendation aims to recommend items to the user by modeling their past behavior sequences and characterize their dynamic interests [24, 39, 42, 46]. Earlier approaches in this field are based on nearest-neighbor methods [12, 21], factorization machine-based methods [44] and Markov Chains [15]. In recent years the advances of deep learning also led to the deployment of many deep sequential recommendation models including CNN-based models [50, 67], RNN-based models [18, 65] and self-attention based models [10, 24, 49]. SASRec [24] and BERT4Rec [49] both utilize the transformer architecture [53] to model correlations among context information in SR. Recently, many works focused on using contrastive self-supervised learning (SSL) to enhance the mutual information between positive samples while increasing the discrimination of negatives [38, 41, 51, 63, 73].

7.2.2. User Intent for Recommendation

In recent times an increasing body of work studied users' intents for improving sequential recommendations [29, 30]. Works in session-based recommendation learn different purchase purposes via a mixture-channel purpose routing network [57], use a multi-intent translation graph neural network to mine user intents [35] or employ a dual-intent network to recommend new items [22]. Work in [71] proposes an attention mixture network based on user intents to achieve multi-level reasoning over item transitions. Another area of research focuses on understanding the sequential patterns in users' interaction behaviors over longer periods. DSSRec [36] introduces a seq2seq training strategy that utilizes multiple future interactions as supervision and incorporates an intent variable derived from both the user's past and future behavior sequences. In ICLRec [7] user intents are represented by latent variables and learned via clustering. The learned intents are leveraged into SR models via contrastive SSL to maximize the agreement between the representation of a sequence and its corresponding intent.

7.2.3. Temporal Information, Repeated Consumption

Time-sensitive recommendation considers the temporal information of item interactions as context features or models temporal decay effects of historical interactions via point processes. In TimeSVD++ [62], timestamps are divided into bins and combined with a collaborative-filtering framework. In tensor factorization methods time is viewed as an extra dimension in the user-item interaction matrix [5, 25, 64]. Other works focus on

capturing trends and user-evolving patterns via attention-based temporal modules [8, 9, 43, 66]. Li et al. [31] extend SASRec by modeling the user-specific time intervals in the item sequence. Recently, TGSRec [11] designs a continuous-time bipartite graph, which captures temporal dynamics within the sequential patterns of user-item interactions. Another line of work applies the Hawkes Process framework [14] to model the temporal decay effects of historical interactions [6, 74], which also increases the capability of the model to predict repeating item interactions [4, 19, 54, 55].

Different from previous works, we not only leverage that repeated interactions occur at intent levels but also show that incorporating personal user information is crucial for learning temporal dynamics. Additionally, our model addresses the gaps in the current understanding of user intents, especially in terms of capturing repeated and periodic patterns, modeling user intents through a hypergraph and soft clustering techniques based on user session information, which significantly enhances personalized recommendation performance.

7.3. Preliminaries

7.3.1. Problem Definition and Notations

In sequential interaction scenarios, the observed user-item interaction data is represented by a set of tuples $\{(u, v, t)\}$, indicating that user $u \in \mathcal{U}$ interacted with item $v \in \mathcal{V}$ at timestamp t. The interactions are sorted chronologically to form a user's interaction sequence $I_u^t = [(v_1, t_1), (v_2, t_2) \ldots, (v_n, t_n)]$, where n is the number of interactions of user u until timestamp t. Based on the varying time intervals between interactions, the sequence S_u can be divided into subsequences (or sessions) whenever the time interval between two interactions exceeds a threshold δ (e. g., a day or hour). The resulting session interaction sequence can be represented as $S_u = [s_1^u, s_2^u, \ldots, s_l^u]$, where s_l^u represents the l-th interaction subsequence of user u containing items from $\mathcal{V}$. The objective of sequential recommendation is to predict the item from the item set $\mathcal{V}$ that the user u is most likely to interact with at a given timestamp t, given their sequence S_u.

7.3.2. Hawkes Processes for Sequential Modeling

A temporal point process is a stochastic process consisting of discrete events localized in the continuous-time domain. In sequential recommendation, the times at which a user interacts with a specific item can be represented as a series of historical events $H_t = [t_1, t_2, \ldots, t_n]$. To model the time of the next event based on previous events, a conditional intensity function $\lambda(t|H_t)$ is introduced. This function represents a stochastic model for the occurrence of the next event given all previous event times and thereby,

affects the characteristics of the temporal point process. In Hawkes Processes [14], the intensity function takes the form of

$$\lambda(t) = \lambda_{Base} + \alpha \sum_{t_i<t} \varphi(t - t_i), \tag{7.1}$$

where λ_{Base} represents the base intensity and each historical event has a self-exciting effect on the current intensity controlled by the triggering kernel φ which determines how each past event boosts the event intensity over time. The parameter α determines the degree of excitation. In the context of sequential recommendation, the base intensity represents the user's basic interest in a target item, and the self-exciting term indicates the cumulative impact of historical interactions on the user's interest over time.

7.4. Proposed Method (HyperHawkes)

As illustrated in Figure 7.3 our HyperHawkes model consists of several major components, including the intent-based global item graph, and a hypergraph-based aggregation layer to generate intent-based item representations for the soft clustering component. The clustered intent-based item representations serve as inputs to the temporal module, which captures users' long-term interests. To model short-term interest we employ an attention-mixture network and combine both long-term and short-term signals in the final prediction layer. In the following, we will detail each component.

7.4.1. Intent-based Hypergraph Network

As user intents are latent by definition and hence are difficult to extract, we propose to induce structural bias via hypergraph modeling to support the underlying soft clustering process to find useful intent representations. Compared to a simple graph with an adjacency matrix reflecting the pairwise relationship between two nodes, hyperedges in hypergaphs can connect more than two nodes and are therefore suitable to model user intents, since item interactions on an intent level naturally comprise a set of items. We assume that in each user session, the user interacts with the system based on one or more intents. To build our intent-based global item hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{E} = \{\varepsilon_i\}$ being the set of hyperedges, we apply the following procedure: First, we extract data-driven user intents as frequently occurring item sets across all training user sessions with a length $>= 2$ via the FP-Max algorithm [13], where the minimum support is set to γ. The threshold parameter γ filters for reliable user intents and drops noisy intents not supported by many other user sessions [38]. For each of the extracted intents, we connect all the corresponding items via a hyperedge $\varepsilon_i \in \mathcal{E}$ to build our global hypergraph. Each hyperedge ε_i has a weight w_i attached, indicating the frequency of the extracted intent in the dataset.

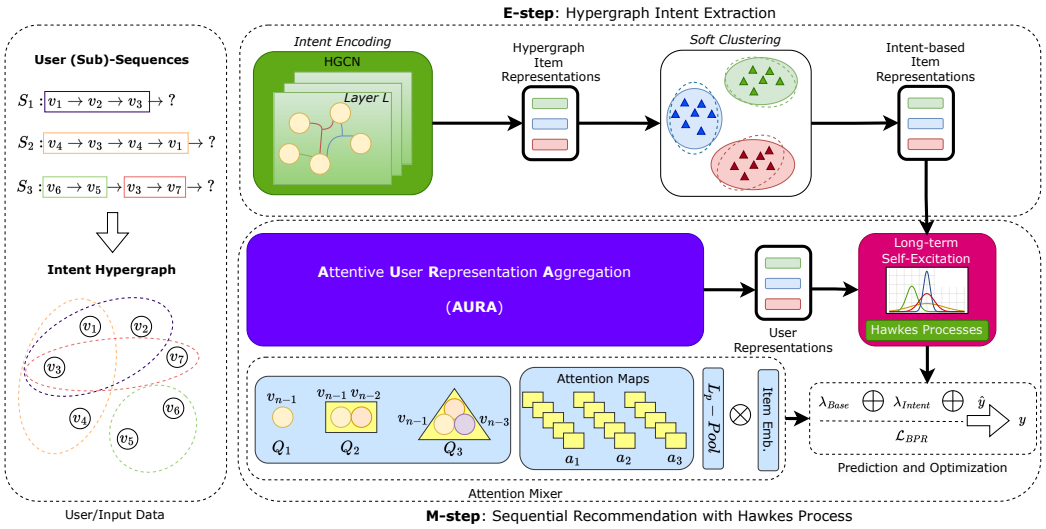


Figure 7.3.: Overall architecture of HyperHawkes: In the E-step of the EM algorithm, our approach extracts latent intent representations via soft clustering of hypergraph-based item embeddings. In the M-step, we compute long-term user preference scores via Hawkes Processes based on the user base excitation from an attentive FISM and self-exciting effects of intents. We fuse short-term scores from the attention-mixture network and the long-term scores to get the preference score of the user for an item.

To generate intent-based global item representations we design a simple hypergraph aggregation layer. For the item v_i's base embedding $\mathbf{x}_i^{(0)}$, we map its corresponding identifier into a dense embedding vector $\mathbf{h}_{v_i} \in \mathbb{R}^d$, where d indicates the dimension. To aggregate information from neighboring nodes we employ the following hypergraph convolution with symmetric normalization in our HGCN component:

$$\mathbf{X}^{(l+1)} = \mathbf{D}^{-1}\mathbf{H}\mathbf{W}\mathbf{B}^{-1}\mathbf{H}^{\top}\mathbf{X}^{(l)}, \tag{7.2}$$

where $\mathbf{H}$ is the incidence matrix, $\mathbf{W}$ is the diagonal hyperedge weight matrix, and $\mathbf{D}$ and $\mathbf{B}$ are the corresponding degree matrices. Compared to the hypergraph convolution presented in [3] we do not make use of learnable weights and a non-linear activation function, since these components are not essential for recommender systems [59, 61]. To combine node embeddings over multiple layers and increase the receptive field of a node we average the node embeddings over L layers to get the final intent-based hypergraph item representations:

$$\mathbf{X}^{(L)} = \frac{1}{L+1}\sum_{l=0}^{L}\mathbf{X}^{l}. \tag{7.3}$$

7.4.2. Intent Representation Learning

On the user interaction sequence level, it is easily observed that user sessions exhibit multiple, dynamically shifting intents, where items can also belong to more than one specific intent alone [7, 48]. Additionally, these intents are not confined solely to individual sessions but are also prevalent among users with similar preferences. Therefore, directly utilizing session representation distributions for intent representations will result in a loss of information. To mitigate this, we introduce a soft clustering component to disentangle latent intents and effectively cluster items to intents.

For our soft clustering component we adopt a soft version of the Lloyd's k-means algorithm [58]. Let $\mathbf{x}_j$ represent the intent-based hypergraph representation $\mathbf{x}_j^{(L)}$ of item v_j and μ_k represent the center of intent cluster k. The variable r_{jk} denotes the probability to which item v_j is assigned to intent cluster k. In the standard k-means algorithm, this assignment is binary, but we relax it to allow fractional values such that $\sum_k r_{jk} = 1$ for all j. Specifically, we define

$$r_{jk} = \frac{\exp(-\beta\|\mathbf{x}_j - \mu_k\|)}{\sum_\ell \exp(-\beta\|\mathbf{x}_j - \mu_\ell\|)}, \tag{7.4}$$

which provides a soft-min assignment of each point to the cluster centers based on distance. We use negative cosine similarity as a distance norm $\|\cdot\|$. Here, β is an inverse-temperature hyperparameter; taking $\beta \to \infty$ recovers the standard k-means assignment. The intent cluster centers can be optimized via an iterative process similar to the traditional k-means updates by alternately setting

$$\mu_k = \frac{\sum_j r_{jk}\mathbf{x}_j}{\sum_j r_{jk}} \quad \forall k = 1, \ldots, K \tag{7.5}$$

$$r_{jk} = \frac{\exp(-\beta\|\mathbf{x}_j - \mu_k\|)}{\sum_\ell \exp(-\beta\|\mathbf{x}_j - \mu_\ell\|)} \quad \forall k = 1, \ldots, K,\ j = 1, \ldots, n. \tag{7.6}$$

These iterations converge to a fixed point where μ remains unchanged between successive updates. Thus, we have soft intent cluster assignments for each item $\mathbf{p}_j \in \mathbf{P}$ corresponding to probabilities that item v_j belongs to one of the intent clusters K. This distribution $\mathbf{p}_j$ serves as the latent intent representation of item v_j.

Since the intent representations $\mathbf{p}_j \in \mathbf{P}$ are latent by definition we face the issue that without the cluster representations, we cannot estimate our model parameters θ and without θ we are not able to find a result for the soft cluster assignment probabilities $\mathbf{P}$. It has been shown that a generalized Expectation-Maximization (EM) framework can resolve this situation [7, 34]. In its basic idea, EM starts with an initial guess of θ and estimates the expected values of our cluster assignments $\mathbf{P}$ in the E-step. In the M-step we maximize the objective w.r.t. the model parameters θ given the expected values of $\mathbf{P}$. These steps are repeated until the likelihood cannot increase anymore. For detailed derivations of the EM framework under the sequential recommendation scenario we refer to [7, 34].

7.4.3. Repeated Long-term Intent Consumption

We employ Hawkes Processes to model the temporal dynamics of long-term interactions on intent level. As defined in Equation 7.1 λ_{Base} reflects the long-term base interest of a user in a specific item at a given point in time t, whereas the second part accounts for the self-exciting effects λ_e and can capture repeated intent behaviors. We detail these two components in the following.

User Base Preference

Users often have diverse or even contrastive preferences (e. g., romantic and horror movies). Hence, using a single embedding vector to represent the long-term user interest is a limiting factor [60]. Previous works mitigate this issue by generating a global and non-causal representation of each user interaction sequence. Previous works [23, 33] built the preference representation of a user for an interacted item by a uniform aggregation of the representation of the other items in the interaction sequence. In our approach, we incorporate an attentive user representation aggregation (AURA) to compute the basic strength of the Hawkes Process μ which computes user representations flexibly based on the current target item representation $\mathbf{h}_v$:

$$\lambda_{Base}(u, v) = \mathbf{h}_u + \sum_{j \in I_u \setminus \{v\}} \frac{\exp(\mathbf{h}_j^\intercal \mathbf{h}_v)}{\sum_{j\prime \in I_u \setminus \{v\}} \exp(\mathbf{h}_{j\prime}^\intercal \mathbf{h}_v)} \mathbf{h}_v \tag{7.7}$$

where $\mathbf{h}_u \in \mathbb{R}^d$ defines the latent user representation and is fused with the long-term preference of user u for item v which is a weighted aggregation of the item representations in the user interaction sequence I_u.

Intent Excitation Learning

The trigger kernel of the intensity function in the Hawkes Process captures the changing excitation over time. Our goal is to leverage the time dynamics of a user's next intent and how previous intents can trigger subsequent interactions. The Hawkes Process simulates the time dynamics to predict the probability of the next event. In our approach, we consider interaction events with the same underlying intent for self-excitation. Particularly, we define intent excitation learning as follows:

$$\lambda_{Intent}(u, v, t) = \alpha_k \sum_{(v\prime, t\prime) \in I_u^t} I_K(\mathbf{p}_v, \mathbf{p}_{v\prime}) \varphi(t - t\prime) \tag{7.8}$$

where I_K denotes the indicator function which returns 1 if item v and $v\prime$ belong to the same intent cluster and are in different user sessions, otherwise it returns 0. Since we use a soft clustering approach to assign intent cluster probabilities to each item we use the Kullback–Leibler divergence for finding items that correspond to the same intent clusters:

$$I_K(\mathbf{p}_v, \mathbf{p}_{v'}) = \mathbf{p}_v \cdot \log\left(\frac{\mathbf{p}_v}{\mathbf{p}_{v'}}\right) < \delta, \tag{7.9}$$

where δ is a parameter to limit the probability distribution distances per intent cluster assignment. The cluster-related parameter α_k weights the degree of excitation. The temporal kernel function $\varphi(\cdot)$ changes with the time interval $\Delta t = t - t\prime$ between items of the same intent and is defined as:

$$\varphi(\Delta t) = (1 - \pi_k)E(\Delta t|1/\beta_k) + \pi_k N(\Delta t|\mu_k, \sigma_k), \tag{7.10}$$

where we leverage an exponential distribution with intent-based parameter β_k to model short-term intent repeat consumption behavior, which diminishes quickly over time. For long-term repeated behavior we employ a normal distribution with mean μ_k and standard deviation σ_k which are also intent representation-based parameters. Using normal distributions to simulate the user dynamic interest changes captures real-world scenarios like item lifecycles and repeated item consumption behavior [19, 55]. The coefficient π_k balances the two distributions. We learn the corresponding parameters of the distributions $\Theta_{Intent} = \{\alpha_k, \beta_k, \mu_k, \sigma_k, \pi_k\}$ by a non-linear transformation of the user representation $\mathbf{h}_u$, item representation $\mathbf{h}_v$ and the intent representation $\mathbf{p}_v$:

$$\Theta_{Intent} = \mathcal{M}(\mathbf{h}_u||\mathbf{h}_v||\mathbf{p}_v), \tag{7.11}$$

where $\mathcal{M}(\cdot)$ is implemented as a two-layer neural network and $||$ denotes the vector concatenation operation. Compared to previous approaches [19, 55] our distribution parameters are not related to item identifiers, but to the corresponding item and intent representations. Hence, our model learns the temporal dynamics on both, item and intent level, and is able to leverage denser input signals, since the number of intents is usually smaller than the number of items in a dataset. Additionally, the incorporation of the user representation to compute the distribution parameters allows our model to learn user-specific repetition behavior which can vary across intents. For instance, one user buys a new phone including accessories every year whereas another user only buys a new phone if the old one is broken, exhibiting a longer intent cycle phase.

We introduced the base intensity $\lambda_{Base}(u, v)$ as well as the long-term self-excitations $\lambda_{Intent}(u, v, t)$ on intent level. Therefore, we define our final long-term excitation for item v_i:

$$\lambda_i(u, v_i, t) = \lambda_{Base}(u, v_i) + \lambda_{Intent}(u, v_i, t) \tag{7.12}$$

7.4.4. Attention Mixtures for Short-term User Interest

The aforementioned components model the users' long-term interest and repeated consumption behavior based on intents. However, a user intent might be exploratory, or the interest may change dynamically during the session. To capture these short-term user interest dynamics, we employ an attention mixture mechanism, following previous works in session-based and sequential recommendation [52, 71]. Following [71] we generate multi-level intent queries on the groups of last items in a user interaction sequence with

length n by employing the deep sets operation [68] and applying linear transformations per level $m \in M$:

$$\mathbf{Q}_M = \mathbf{W}_M \left(\sum \{\mathbf{h}_{v_i}\}_{i=n,\ldots,n-M+1} \right). \tag{7.13}$$

These generated queries are then used to compute multi-head attention weights as:

$$\alpha_h = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{W}_Q(\mathbf{K}\mathbf{W}_K)^\intercal}{\sqrt{d}} \right), \tag{7.14}$$

where $\mathbf{Q} \in \mathbb{R}^{l \times d}$ is the query matrix, $\mathbf{K} \in \mathbb{R}^{n \times d}$ represents the hidden representation of each item in the sequence and $\mathbf{W}^Q, \mathbf{W}^K \in \mathbb{R}^{d \times d}$ are trainable parameters. We apply L_p-pooling [20] to pool the attention map and multiply the hidden representation of the items in the sequence with the corresponding pooled attention weights to get the final short-term sequence representation $\mathbf{s}_u$.

7.4.5. Prediction and Model Optimization

For the next-item prediction task we need to combine long-term and short-term interests of users. We use the short-term sequence representation $\mathbf{s}_u$ to compute the short-term interest score $\hat{y}_i = \mathbf{s}_u^\intercal \mathbf{h}_{v_i}$, for item v_i. Then, we add the long-term excitation score λ_i and the short-term interest score $\hat{y}_i$ to get the final preference score:

$$y_i = \lambda_i + \hat{y}_i. \tag{7.15}$$

To learn the parameters of our recommendation model in the M-step of the EM algorithm we adopt the pairwise ranking (BPR) loss for optimization as follows:

$$\mathcal{L}_{BPR} = -\sum_{u \in \mathcal{U}} \sum_{i=1}^{N_u} \log \sigma(y_{ui} - y_{uj}), \tag{7.16}$$

where σ denotes the sigmoid function and y_{uj} reflects the preference score of user u to a randomly sampled negative item $j \notin I_u^t$.

7.5. Experiments and Results

In this section, we provide the setup and results of extensive experiments to evaluate our proposed model, where we compare HyperHawkes to various state-of-the-art models in SR. Given our overall goal of investigating the impact of intent repeat-consumption and fusing short- and long-term interests of users, we aim to answer the following research questions:

- *RQ1:* How does our proposed HyperHawkes compare to other state-of-the-art SR methods on different datasets?
- *RQ2:* How do different components in HyperHawkes contribute to the performance?
- *RQ3:* How sensitive is HyperHawkes to different hyperparameter settings (e. g., L, K)?

7.5.1. Experimental Setup

Datasets and Preprocessing

We conduct experiments on four representative datasets from the e-commerce, food delivery, and music domains [17, 27]. The *Ta-Feng*[2] dataset contains Chinese grocery store transaction data from 2001. *SMM*[3] chronicles five months of user behavior from a large online store [47]. For this industrial-scale dataset, we sample 20,000 random users to maintain consistency with the other datasets. The *DHRD* (Delivery Hero Recommendation Dataset)[4] [2] comprises food delivery orders from three distinct cities, encompassing different vendors and dishes; we use the data related to the city of Stockholm. Lastly, the *NowPlaying* dataset includes music listening behavior of users based on Twitter data [70]. It is worth noting, that we do not provide evaluation for the widely used Amazon review datasets [37], the MovieLens datasets[5], or the Yelp review datasets[6], since those datasets are rating/review-based and therefore do not include repeated item consumptions, making them unsuitable for the scenario of repeated intent modeling [17, 27].

Table 7.1.: Dataset statistics (after preprocessing): Number of users, items, interactions, avg. sequence length and sparsity.

Dataset	**Ta-Feng**	**SMM**	**DHRD**	**NowPlaying**
$\lvert\mathcal{U}\rvert$	26,162	12,098	42,774	11,310
$\lvert\mathcal{V}\rvert$	15,642	22,167	20,883	15,905
# Interactions	0.78m	0.87m	0.52m	1.12m
Avg. length	29.99	71.97	12.30	86.39
Sparsity	99.80%	99.67%	99.94%	99.45%

We follow the preprocessing steps as in [7, 19] for the four datasets: We keep the *5-core* datasets, where users and items with less than 5 interactions are filtered out. Table 7.1 provides an overview of the datasets after preprocessing. To split the datasets, we follow common practice in sequential recommendation and use interactions with the second latest time for validation and interactions with the latest timestamp for testing.

[2] https://www.kaggle.com/datasets/chiranjivdas09/ta-feng-grocery-dataset
[3] https://disk.yandex.ru/d/fSEBIQYZusAAuw/datasets/data_smm
[4] https://github.com/deliveryhero/dh-reco-dataset
[5] https://grouplens.org/datasets/movielens
[6] https://www.yelp.com/dataset

Following [17, 28, 69], we use the whole item set without negative sampling to rank the predictions. We adopt *HR@{5,20}* (Hit Ratio) and *NDCG@{5,20}* (Normalized Discounted Cumulative Gain) as evaluation metrics.

Baseline Methods

We compare HyperHawkes with the following representative baseline and state-of-the-art methods for sequential recommendation:

Static models: BPR-MF [45] is a non-sequential model and characterizes the pairwise interactions via matrix factorization.

Standard sequential & Transformer models: We include GRU4Rec [18], an RNN-based method and SASRec [24] as transformer-based baseline method for SR.

Temporal & intent-based models: SLRC [55] is a widely used model and one of the first to model item repeat consumption. It combines matrix factorization with a temporal point process, effectively capturing short-term and product lifetime effects. RepeatNet [43] proposes a novel repeat-explore mechanism to balance repeated and new item consumption. For intent-based methods, we include HIDE [32] which models intents via session hypergraphs. Other state-of-the-art approaches include ICLRec [7] and ICSRec [40], where user intents are learned via clustering and Atten-Mixer [71], where intents are modelled via a multi-level network.

Implementation Details

We rely on the RecBole framework [72] to implement our approach, using the provided baseline models or re-implementing as needed. For all models, the embedding size is set to 64 and the batch size to 512. We do not limit the number of training epochs, but adopt an early-stopping strategy, which stops training after five consecutive rounds of performance decrease on the validation set. Each baseline model is optimized according to its corresponding hyperparameters.

For the optimization of HyperHawkes, we use Adam [26] with a learning rate of 0.001. The number of layers L in the HGCN component and number of intent clusters K are searched in the ranges of $\{1, 2, 3, 4, 5\}$ and $\{2, 4, \ldots, 128\}$ respectively. For the attention mixture network, we search the number of heads in the range of $\{1, 2, 4, 8\}$ and the number of levels M in $\{1, 2, \ldots, 10\}$. The threshold parameters γ and δ are set to 5e−4 and 1e−12. Our implementation is based on PyTorch 1.13.1 and Python 3.8.16, and runs on a workstation with an AMD Ryzen 2950X, a GeForce RTX 2070, and 256 GB main memory. We publish the code and the pre-processed datasets on GitHub: `https://github.com/dbis-uibk/HyperHawkes`.

Table 7.2.: Model performance on all four datasets (± standard deviation for HyperHawkes). All improvements of HyperHawkes over the second best model are significant (paired t-test, $p < .05$), best results are in boldface, second-best results are underlined.

Dataset	Metric	BPR-MF	GRU4Rec	SASRec	SLRC	RepeatNet	HIDE	ICLRec	Atten-Mixer	ICSRec	HyperHawkes	Improv.
Ta-Feng	HR@5	0.0699	0.0657	0.0812	0.0714	0.0432	0.0616	0.0746	0.0878	0.0784	**0.1108**±0.0015	26.19%
	HR@20	0.0943	0.1215	0.1629	0.1284	0.1006	0.0853	0.1415	0.1645	0.1566	**0.1984**±0.0030	20.60%
	NDCG@5	0.0541	0.0459	0.0528	0.0488	0.0307	0.0419	0.0527	0.0605	0.0519	**0.0765**±0.0014	26.44%
	NDCG@20	0.0610	0.0616	0.0761	0.0650	0.0469	0.0485	0.0716	0.0823	0.0742	**0.1015**±0.0015	23.32%
SMM	HR@5	0.0542	0.0586	0.0876	0.1170	0.1291	0.0391	0.0526	0.0817	0.0686	**0.1483**±0.0019	14.87%
	HR@20	0.1056	0.1323	0.1687	0.1853	0.1968	0.0781	0.1101	0.1638	0.1505	**0.2444**±0.0009	24.19%
	NDCG@5	0.0373	0.0393	0.0606	0.0840	0.0972	0.0272	0.0357	0.0561	0.0427	**0.1018**±0.0002	4.73%
	NDCG@20	0.0516	0.0602	0.0836	0.1037	0.1175	0.0383	0.0520	0.0793	0.0656	**0.1294**±0.0004	10.13%
DHRD	HR@5	0.2156	0.1439	0.2065	0.2775	0.2702	0.1878	0.2554	0.2211	0.2129	**0.2982**±0.0055	7.45%
	HR@20	0.3805	0.3214	0.4651	0.4158	0.3211	0.2625	0.4544	0.4295	0.4715	**0.4830**±0.0019	2.43%
	NDCG@5	0.1488	0.0946	0.1303	0.2031	0.1983	0.1356	0.1740	0.1489	0.1323	**0.2089**±0.0031	2.85%
	NDCG@20	0.1963	0.1450	0.2039	0.2430	0.2142	0.1572	0.2312	0.2084	0.2145	**0.2621**±0.0069	7.86%
NowPlaying	HR@5	0.1272	0.0992	0.1229	0.1756	0.1765	0.1079	0.1654	0.1475	0.1375	**0.1842**±0.0011	4.36%
	HR@20	0.2730	0.2327	0.2715	0.3117	0.2996	0.1984	0.3135	0.3011	0.2931	**0.3526**±0.0006	12.47%
	NDCG@5	0.0879	0.0650	0.0802	0.1197	0.1217	0.0787	0.1156	0.1028	0.0929	**0.1242**±0.0007	2.05%
	NDCG@20	0.1289	0.1025	0.1221	0.1589	0.1581	0.1042	0.1574	0.1462	0.1368	**0.1713**±0.0008	8.43%

7.5.2. Performance Comparison (RQ1)

In Table 7.2 we compare the performance of HyperHawkes and the baselines. Interestingly, BPR-MF performs competitively with GRU4Rec and SASRec, contrasting the general assumption that sequential models generally outperform non-sequential methods. This displays the importance of learning temporal dynamics of repeated user behavior and the incorporation of user intent.

Advanced time-sensitive sequential models often incorporate additional temporal signals to augment recommendation performance. For instance, TiSASRec integrates both the item positions and time intervals in a sequence, yielding superior performance than its transformer-based counterpart SASRec. We further observe that leveraging contrastive SSL in transformer-based architectures can improve performance, as exhibited by ICLRec which optimizes sequence representations via contrastive SSL at the user intent level. Also, the other intent-based method Atten-Mixer shows significant performance gains over standard sequential models. Among the baseline methods, SLRC and RepeatNet exhibit improved performance even over more sophisticated temporal and intent-based models, underpinning their robustness in recommendation tasks and their ability to model item repeat consumption.

HyperHawkes triumphs over all other methods across all datasets, marking a significant advancement. The average improvements compared with the best baseline per dataset range from 2.43% to 24.19% in HR@20 and from 7.86% to 23.32% in NDCG@20. We attribute this increase in performance to the ability of our approach to effectively model long-term intent repeat behavior and short-term user interest, which we show in detail in our ablation study.

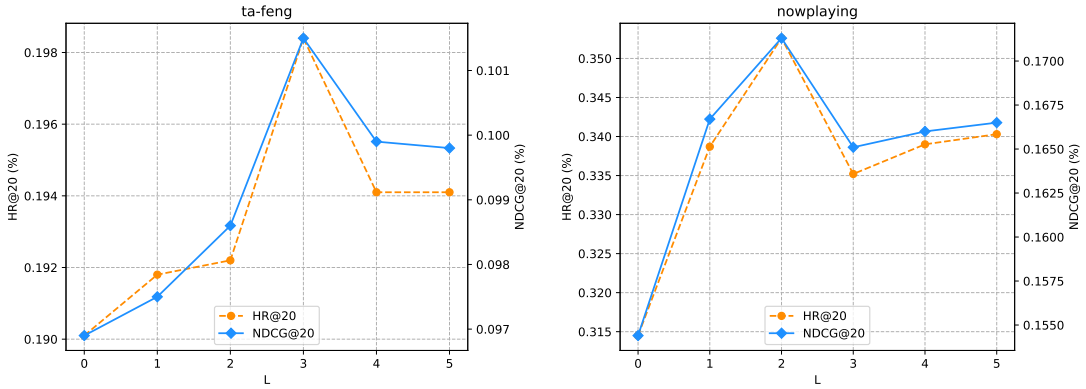
In terms of efficiency and model complexity, we report the training time per epoch on the *Ta-Feng* dataset as a practical proxy for model complexity. Intent-based models like HIDE, ICLRec, Atten-Mixer and ICSRec require 2231.63, 254.21, 13.10 and 174.17 seconds/epoch, respectively. SLRC and RepeatNet, focusing on repeat consumption, need 13.31s and 29.64s. HyperHawkes takes 27.75s per epoch on training and therefore, is more efficient than most of the other sequential models, while substantially outperforming these models in recommendation performance. A similar trend in model complexity is also seen for the other datasets.

7.5.3. Ablation Study (RQ2)

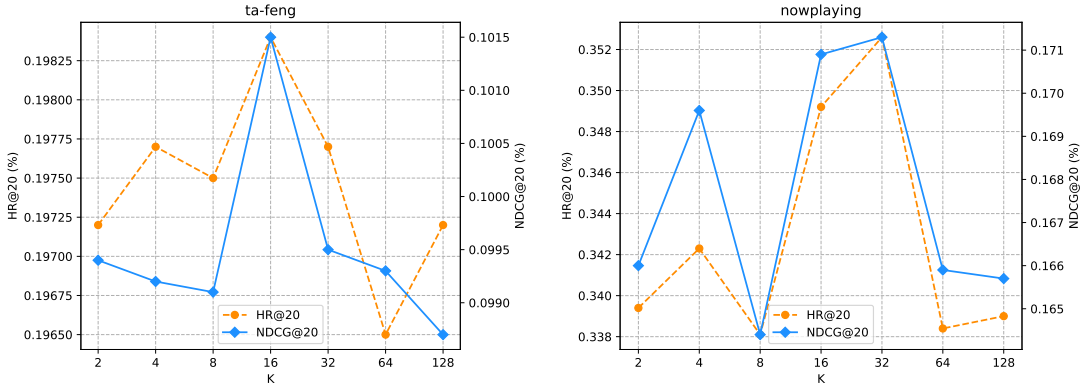
Table 7.3.: Ablation study of HyperHawkes. The symbol ↓ indicates a performance drop of more than 10%, ND=NDCG.

Model	Ta-Feng		NowPlaying	
	HR@20	ND@20	HR@20	ND@20
(A) w/o LT-SINE	0.1632↓	0.0842↓	0.3331	0.1637
(B) w/o LT-UE	0.1818	0.0911↓	0.3241	0.1570
(C) w/o HGCN	0.1901	0.0969	0.3145↓	0.1544↓
(D) w/o SC	0.1732↓	0.0867↓	0.3377	0.1666
(E) only ST-ATM	0.1668↓	0.0841↓	0.2954↓	0.1451↓
(F) w/o ST-ATM	0.0914↓	0.0558↓	0.3314	0.1625
HyperHawkes	**0.1984**	**0.1015**	**0.3526**	**0.1713**

HyperHawkes contains several components: a hypergraph-based graph convolutional network (HGCN), soft clustering (SC), user base interest (LT-UE), intent excitation learning (LT-SINE), and a short-term attention mixture network (ST-ATM). To verify the effectiveness of each component, we conduct an ablation study on two datasets and show the results in Table 7.3. The *Ta-Feng* and *NowPlaying* datasets were chosen due to their different domains and characteristics in terms of repeat consumption (e. g., e-commerce vs. music streaming). From (A) and (B) we can see the impact of different components in the Hawkes Process for modeling temporal dynamics. Eliminating the intent excitation learning (A) or the user base preference (B), notably diminishes the performance to a similar extent. This shows the importance of extracting latent intents and modeling repeat behavior on the intent level compared to the item level only. We also investigate the effect of our proposed hypergraph-based network in (C), where removing the component also leads to a significant performance drop. This backs our assumption that inducing structural bias through the HGCN supports the soft clustering process and leads to more representative cluster/intent representations. Similar effects can be observed when dropping the soft clustering component in (D) and using a standard k-means instead, which showcases the benefit of disentangling user intents via soft probability distributions. Lastly, we explore the effects of the short-term attention mixture



(a) Different number of layers L in our HGCN component.



(b) Number of intent clusters parameter K.

Figure 7.4.: Impact of hyper-parameters in HyperHawkes.

network. Relying only on the short-term component without any long-term effects (E) results in a noticeable performance drop. Dropping the short-term component (F) shows a substantial decline compared to the full model, reflecting the critical role of short-term user behavior understanding. The incorporation of both short-term and long-term effects leads to the best overall performance. The ablation study results for the other two datasets *SMM* and *DHRD* are consistent with these findings, but are not reported due to space constraints.

7.5.4. Impact of Hyper-Parameters (RQ3)

In this section, we investigate the impact of different hyper-parameters. We focus on the number of layers L in the HGCN and the number of intent clusters K, since these hyper-parameters are related to the intent excitation learning, which has shown to have the highest impact on the performance of the final model (see Section 7.5.3). Figure 7.4a

shows the performance of our model with different settings of layers L on the *Ta-Feng* and *NowPlaying* datasets. A higher number of layers in the hypergraph-based network does not necessarily lead to an increase in performance due to oversmoothing, where node representations converge to the same values. We can find a sweet spot layer setting L of 3 (*Ta-Feng*) and 2 (*NowPlaying*).

Our main contribution lies in the temporal modeling of user intents, extracted by soft clustering. This requires pre-defining the number of clusters K before training. Due to dataset heterogeneity, K needs to be tuned to each dataset's characteristics. Figure 7.4b shows the performance differences with different cluster counts. The optimal setting differs by dataset, with *Ta-Feng* performing best at 16 clusters and *NowPlaying* at 32 clusters.

7.6. Conclusion

We proposed HyperHawkes, a novel Hypergraph-based Hawkes Process model to comprehensively model temporal dynamics of user intents for generating personalized sequential recommendations. We extract intent representations via soft clustering of hypergraph-based item representations. Our model learns the long-term excitation of intents and items via Hawkes Processes and models short-term interests of users via a custom attention mixture component. The fused user preference scores from the long-term and short-term components enable temporal and personalized recommendations. Cluster discovery and learning temporal dynamics are alternately optimized under a generalized EM framework. Our extensive experimental results on four datasets demonstrate the effectiveness of HyperHawkes, outperforming all other state-of-the-art methods. The ablation study showed that modeling repeat consumption is more important than focusing on short-term interest shifts of users.

References

[1] A. Anderson, R. Kumar, A. Tomkins, and S. Vassilvitskii. The dynamics of repeat consumption. In *23rd International World Wide Web Conference, WWW '14*, pages 419–430. ACM, 2014.

[2] Y. Assylbekov, R. Bali, L. Bovard, and C. Klaue. Delivery hero recommendation dataset: A novel dataset for benchmarking recommendation algorithms. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, pages 1042–1044. ACM, 2023.

[3] S. Bai, F. Zhang, and P. H. S. Torr. Hypergraph convolution and hypergraph attention. *Pattern Recognit.*, 110:107637, 2021.

[4] T. Bai, L. Zou, W. X. Zhao, P. Du, W. Liu, J. Nie, and J. Wen. Ctrec: A long-short demands evolution model for continuous-time recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019*, pages 675–684. ACM, 2019.

[5] P. Bhargava, T. Phan, J. Zhou, and J. Lee. Who, what, when, and where: multi-dimensional collaborative recommendations using tensor factorization on sparse user-generated data. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015*, pages 130–140. ACM, 2015.

[6] R. Cai, X. Bai, Z. Wang, Y. Shi, P. Sondhi, and H. Wang. Modeling sequential online interactive behaviors with temporal point process. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018*, pages 873–882. ACM, 2018.

[7] Y. Chen, Z. Liu, J. Li, J. J. McAuley, and C. Xiong. Intent contrastive learning for sequential recommendation. In *WWW '22: The ACM Web Conference 2022*, pages 2172–2182. ACM, 2022.

[8] J. Cho, D. Hyun, S. Kang, and H. Yu. Learning heterogeneous temporal patterns of user preference for timely recommendation. In *WWW '21: The Web Conference 2021*, pages 1274–1283. ACM, 2021.

[9] S. M. Cho, E. Park, and S. Yoo. MEANTIME: mixture of attention mechanisms with multi-temporal embeddings for sequential recommendation. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems*, pages 515–520. ACM, 2020.

[10] G. de Souza Pereira Moreira, S. Rabhi, J. M. Lee, R. Ak, and E. Oldridge. Transformers4rec: bridging the gap between NLP and sequential / session-based recommendation. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems*, pages 143–153. ACM, 2021.

[11] Z. Fan, Z. Liu, J. Zhang, Y. Xiong, L. Zheng, and P. S. Yu. Continuous-time sequential recommendation with temporal graph collaborative transformer. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*, pages 433–442. ACM, 2021.

[12] D. Garg, P. Gupta, P. Malhotra, L. Vig, and G. Shroff. Sequence and time aware neighborhood for session-based recommendations: STAN. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019*, pages 1069–1072. ACM, 2019.

[13] G. Grahne and J. Zhu. Efficiently using prefix-trees in mining frequent itemsets. In *FIMI '03, Frequent Itemset Mining Implementations, Proceedings of the ICDM 2003 Workshop on Frequent Itemset Mining Implementations*, volume 90 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2003.

[14] A. G. Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

[15] R. He and J. J. McAuley. Fusing similarity models with markov chains for sparse sequential recommendation. In *IEEE 16th International Conference on Data Mining, ICDM 2016*, pages 191–200. IEEE Computer Society, 2016.

[16] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017*, pages 173–182. ACM, 2017.

[17] B. Hidasi and Á. T. Czapp. Widespread flaws in offline evaluation of recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, pages 848–855. ACM, 2023.

[18] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, Conference Track Proceedings*, 2016.

[19] C. Huang, S. Wang, X. Wang, and L. Yao. Modeling temporal positive and negative excitation for sequential recommendation. In *Proceedings of the ACM Web Conference 2023, WWW 2023*, pages 1252–1263. ACM, 2023.

[20] A. Hyvärinen and U. Köster. Complex cell pooling and the statistics of natural images. *Network: Computation in Neural Systems*, 18(2):81–100, 2007.

[21] D. Jannach and M. Ludewig. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017*, pages 306–310. ACM, 2017.

[22] D. Jin, L. Wang, Y. Zheng, G. Song, F. Jiang, X. Li, W. Lin, and S. Pan. Dual intent enhanced graph neural network for session-based new item recommendation. In *Proceedings of the ACM Web Conference 2023, WWW 2023*, pages 684–693. ACM, 2023.

[23] S. Kabbur, X. Ning, and G. Karypis. FISM: factored item similarity models for top-n recommender systems. In *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013*, pages 659–667. ACM, 2013.

[24] W. Kang and J. J. McAuley. Self-attentive sequential recommendation. In *IEEE International Conference on Data Mining, ICDM 2018*, pages 197–206. IEEE Computer Society, 2018.

[25] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010*, pages 79–86. ACM, 2010.

[26] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*, 2015.

[27] A. Klenitskiy, A. Volodkevich, A. Pembek, and A. Vasilev. Does it look sequential? an analysis of datasets for evaluation of sequential recommendations. In *Proceedings of the 18th ACM Conference on Recommender Systems*, RecSys '24, pages 1067–1072. ACM, 2024.

[28] W. Krichene and S. Rendle. On sampled metrics for item recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1748–1757. ACM, 2020.

[29] C. Li, Z. Liu, M. Wu, Y. Xu, H. Zhao, P. Huang, G. Kang, Q. Chen, W. Li, and D. L. Lee. Multi-interest network with dynamic routing for recommendation at tmall. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 2615–2623, 2019.

[30] H. Li, X. Wang, Z. Zhang, J. Ma, P. Cui, and W. Zhu. Intention-aware sequential recommendation with structured intent transition. *IEEE Trans. Knowl. Data Eng.*, 34(11):5403–5414, 2022.

[31] J. Li, Y. Wang, and J. J. McAuley. Time interval aware self-attention for sequential recommendation. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining*, pages 322–330. ACM, 2020.

[32] Y. Li, C. Gao, H. Luo, D. Jin, and Y. Li. Enhancing hypergraph neural networks with intent disentanglement for session-based recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1997–2002. ACM, 2022.

[33] J. Lin, W. Pan, and Z. Ming. FISSA: fusing item similarity models with self-attention networks for sequential recommendation. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems*, pages 130–139. ACM, 2020.

[34] Z. Lin, C. Tian, Y. Hou, and W. X. Zhao. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In *WWW '22: The ACM Web Conference 2022*, pages 2320–2329. ACM, 2022.

[35] Z. Liu, X. Li, Z. Fan, S. Guo, K. Achan, and P. S. Yu. Basket recommendation with multi-intent translation graph neural network. In *2020 IEEE International Conference on Big Data, IEEE BigData 2020*, pages 728–737. IEEE, 2020.

[36] J. Ma, C. Zhou, H. Yang, P. Cui, X. Wang, and W. Zhu. Disentangled self-supervision in sequential recommenders. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 483–491. ACM, 2020.

[37] J. J. McAuley, C. Targett, Q. Shi, and A. van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52. ACM, 2015.

[38] A. Peintner, A. R. Mohammadi, and E. Zangerle. SPARE: shortest path global item relations for efficient session-based recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, pages 58–69. ACM, 2023.

[39] A. Peintner, M. Moscati, E. Parada-Cabaleiro, M. Schedl, and E. Zangerle. Unsupervised graph embeddings for session-based recommendation with item features. In *CARS: Workshop on Context-Aware Recommender Systems (RecSys '22)*, 2022.

[40] X. Qin, H. Yuan, P. Zhao, G. Liu, F. Zhuang, and V. S. Sheng. Intent contrastive learning with cross subsequences for sequential recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining, WSDM 2024*, pages 548–556. ACM, 2024.

[41] R. Qiu, Z. Huang, H. Yin, and Z. Wang. Contrastive learning for representation degeneration problem in sequential recommendation. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining*, pages 813–823. ACM, 2022.

[42] M. Quadrana, P. Cremonesi, and D. Jannach. Sequence-aware recommender systems. *ACM Computing Surveys (CSUR)*, 51(4):1–36, 2018.

[43] P. Ren, Z. Chen, J. Li, Z. Ren, J. Ma, and M. de Rijke. Repeatnet: A repeat aware neural recommendation machine for session-based recommendation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 4806–4813. AAAI Press, 2019.

[44] S. Rendle. Factorization machines. In *2010 IEEE International conference on data mining*, pages 995–1000. IEEE, 2010.

[45] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461. AUAI Press, 2009.

[46] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010*, pages 811–820. ACM, 2010.

[47] V. Shevchenko, N. Belousov, A. Vasilev, V. Zholobov, A. Sosedka, N. Semenova, A. Volodkevich, A. Savchenko, and A. Zaytsev. From variability to stability: advancing recsys benchmarking practices. *arXiv preprint arXiv:2402.09766*, 2024.

[48] J. Su, C. Chen, W. Liu, F. Wu, X. Zheng, and H. Lyu. Enhancing hierarchy-aware graph networks with deep dual clustering for session-based recommendation. In *Proceedings of the ACM Web Conference 2023*, pages 165–176, 2023.

[49] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang. Bert4rec: sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019*, pages 1441–1450. ACM, 2019.

[50] J. Tang and K. Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018*, pages 565–573. ACM, 2018.

[51] C. Tian, Z. Lin, S. Bian, J. Wang, and W. X. Zhao. Temporal contrastive pretraining for sequential recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 1925–1934. ACM, 2022.

[52] V. Tran, G. Salha-Galvan, B. Sguerra, and R. Hennequin. Attention mixtures for time-aware sequential recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023*, pages 1821–1826. ACM, 2023.

[53] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 5998–6008, 2017.

[54] C. Wang, M. Zhang, W. Ma, Y. Liu, and S. Ma. Make it a chorus: knowledge- and time-aware item modeling for sequential recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, pages 109–118. ACM, 2020.

[55] C. Wang, M. Zhang, W. Ma, Y. Liu, and S. Ma. Modeling item-specific temporal dynamics of repeat consumption for recommender systems. In *The World Wide Web Conference, WWW 2019*, pages 1977–1987. ACM, 2019.

[56] J. Wang, R. Louca, D. Hu, C. Cellier, J. Caverlee, and L. Hong. Time to shop for valentine's day: shopping occasions and sequential recommendation in e-commerce. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining*, pages 645–653. ACM, 2020.

[57] S. Wang, L. Hu, Y. Wang, Q. Z. Sheng, M. Orgun, and L. Cao. Modeling multi-purpose sessions for next-item recommendations via mixture-channel purpose routing networks. In *International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence, 2019.

[58] B. Wilder, E. Ewing, B. Dilkina, and M. Tambe. End to end learning and optimization on graphs. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 4674–4685, 2019.

[59] F. Wu, A. H. S. Jr., T. Zhang, C. Fifty, T. Yu, and K. Q. Weinberger. Simplifying graph convolutional networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97 of *Proceedings of Machine Learning Research*, pages 6861–6871. PMLR, 2019.

[60] L. Wu, S. Li, C. Hsieh, and J. Sharpnack. SSE-PT: sequential recommendation via personalized transformer. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems*, pages 328–337. ACM, 2020.

[61] X. Xia, H. Yin, J. Yu, Y. Shao, and L. Cui. Self-supervised graph co-training for session-based recommendation. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*, pages 2180–2190. ACM, 2021.

[62] L. Xiang and Q. Yang. Time-dependent models in collaborative filtering based recommender system. In *2009 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2009*, pages 450–457. IEEE Computer Society, 2009.

[63] X. Xie, F. Sun, Z. Liu, S. Wu, J. Gao, J. Zhang, B. Ding, and B. Cui. Contrastive learning for sequential recommendation. In *38th IEEE International Conference on Data Engineering, ICDE 2022*, pages 1259–1273. IEEE, 2022.

[64] L. Xiong, X. Chen, T. Huang, J. G. Schneider, and J. G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proceedings of the SIAM International Conference on Data Mining, SDM 2010*, pages 211–222. SIAM, 2010.

[65] C. Xu, P. Zhao, Y. Liu, J. Xu, V. S. Sheng, Z. Cui, X. Zhou, and H. Xiong. Recurrent convolutional neural network for sequential recommendation. In *The World Wide Web Conference, WWW 2019*, pages 3398–3404. ACM, 2019.

[66] W. Ye, S. Wang, X. Chen, X. Wang, Z. Qin, and D. Yin. Time matters: sequential recommendation with complex temporal information. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, pages 1459–1468. ACM, 2020.

[67] F. Yuan, A. Karatzoglou, I. Arapakis, J. M. Jose, and X. He. A simple convolutional generative network for next item recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019*, pages 582–590. ACM, 2019.

[68] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Póczos, R. Salakhutdinov, and A. J. Smola. Deep sets. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 3391–3401, 2017.

[69] E. Zangerle and C. Bauer. Evaluating Recommender Systems: Survey and Framework. *ACM Computing Surveys*, 55(8):170:1–170:38, 2022.

[70] E. Zangerle, M. Pichl, W. Gassler, and G. Specht. #nowplaying music dataset: extracting listening behavior from twitter. In *Proceedings of the First International Workshop on Internet-Scale Multimedia Management, WISMM 2014*, pages 21–26. ACM, 2014.

[71] P. Zhang, J. Guo, C. Li, Y. Xie, J. Kim, Y. Zhang, X. Xie, H. Wang, and S. Kim. Efficiently leveraging multi-level user intent for session-based recommendation via atten-mixer network. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM 2023*, pages 168–176. ACM, 2023.

[72] W. X. Zhao, S. Mu, Y. Hou, Z. Lin, Y. Chen, X. Pan, K. Li, Y. Lu, H. Wang, C. Tian, Y. Min, Z. Feng, X. Fan, X. Chen, P. Wang, W. Ji, Y. Li, X. Wang, and J. Wen. Recbole: towards a unified, comprehensive and efficient framework for recommendation algorithms. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*, pages 4653–4664. ACM, 2021.

[73] K. Zhou, H. Wang, W. X. Zhao, Y. Zhu, S. Wang, F. Zhang, Z. Wang, and J. Wen. S3-rec: self-supervised learning for sequential recommendation with mutual information maximization. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management*, pages 1893–1902. ACM, 2020.

[74] Y. Zhu, H. Li, Y. Liao, B. Wang, Z. Guan, H. Liu, and D. Cai. What to do next: modeling user behaviors by time-lstm. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*, pages 3602–3608. ijcai.org, 2017.

8. Nuanced Music Emotion Recognition via Semi-Supervised Multi-Relational Graph Neural Network

Publications

A. Peintner, M. Moscati, Y. Kinoshita, R. Vogl, P. Knees, M. Schedl, H. Strauss, M. Zentner, and E. Zangerle. Nuanced music emotion recognition via a semi-supervised multi-relational graph neural network. *Transactions of the International Society for Music Information Retrieval*, 8(1):140–153, 2025. DOI: 10.5334/tismir.235

Abstract

Music Emotion Recognition (MER) seeks to understand the complex emotional landscapes elicited by music, acknowledging music's profound social and psychological roles beyond traditional tasks such as genre classification or content similarity. MER relies heavily on high-quality emotional annotations, which provide the foundation for training models to recognize emotions. However, collecting these annotations is both complex and costly, leading to limited availability of large-scale datasets for MER. Recent works in MER for automatically extracting emotion aim to learn track representations in a supervised manner. However, these approaches mainly utilize simpler emotion models due to limited datasets or the lack of necessity of sophisticated emotion models and ignore hidden inter-track relations, which are beneficial for a semi-supervised learning setting. This paper proposes a novel approach to MER by constructing a multi-relational graph that encapsulates different facets of music. We leverage Graph Neural Networks (GNNs) to model intricate inter-track relationships and capture structurally induced representations from user data, such as listening histories, genres and tags. Our model, the Semi-supervised Multi-relational Graph Neural Network for Emotion Recognition (SRGNN-Emo), innovates by combining graph-based modeling with semi-supervised learning, using rich user data to extract nuanced emotional profiles from music tracks. Through extensive experimentation, SRGNN-Emo achieves significant improvements in R^2 and RMSE metrics for predicting the intensity of nine continuous emotions (GEMS), demonstrating its superior capability in capturing and predicting complex emotional expressions in music.

8.1. Introduction

Music's ability to express and evoke emotions is a universally acknowledged phenomenon, transcending cultural and linguistic barriers. It plays a pivotal role in human experience, offering a medium through which emotions can be articulated, shared, and understood. This unique capacity of music to convey a wide range of emotional states makes it a subject of considerable interest in the interdisciplinary fields of psychology, neuroscience, and musicology [24, 54]. Particularly, Music Emotion Recognition (MER) is a computational task aimed at automatically identifying the emotional expressions contained within music or the emotions elicited in listeners by music [50]. MER researchers rely on a collection of datasets, where the amount of annotated tracks per dataset is rather small [3, 55]. This is unsurprising since collecting high-quality emotional annotations of tracks is complex and expensive [41]. While small-scale datasets are valuable for MER advancements [28], for music retrieval and recommendation tasks, it is inevitable to have access to a large catalog of tracks annotated with emotion labels, especially in the context of personalized music retrieval [52]. An alternative method for gathering emotional data in music involves extracting emotions from user tags. These tags are readily accessible and available on a large scale. However, they often contain noise and personal bias, and they lack the depth and quality that set apart expert-annotated data. Such expert data is typically collected through user studies informed by psychological principles [28, 33].

There are several approaches to tackle MER aiming to tag tracks with corresponding emotion labels or profiles. Textual information is one of the data type employed in assignments that incorporate emotion labels, as evidenced by numerous studies [21, 22, 53]. Specifically, when undertaking emotion recognition based on music data, lyrics frequently serve as the primary source of input [12, 13]. A different body of research highlights the significant role of acoustic features in emotion recognition tasks [16, 34, 51, 52]. This perspective sheds light on the complexity of musical emotion, suggesting that the emotional content of music cannot be fully captured through lyrics alone. The recognition that both modalities, textual and acoustic, play a critical role in the perception and interpretation of musical emotions is well known in the scientific community [16, 38, 49].

Most of the aforementioned approaches perform classification for emotion labels per track or employ basic or categorical emotion models (e. g., arousal and valence) in a supervised learning setting, which often fail to capture the richness and variability of musical emotions [13, 52]. In contrast, this work draws on a domain-specific model, specifically devised to account for the richness of emotions induced by music [54]. Starting with 515 emotion terms, [54] successively eliminated those terms that were rarely used to describe music-evoked emotions and retained a few dozen core emotion terms, titled GEMS for Geneva Emotional Music Scale. GEMS is hierarchically organized in three second-order and nine first-order factors as shown in Figure 8.1. These factors are: (1) vitality (power and joyful activation), (2) sublimity (wonder, transcendence, tenderness, nostalgia, and peacefulness), (3) unease (tension and sadness). An additional distinctive feature of the GEMS is that it accounts not only for perceived emotion but also, and in particular,

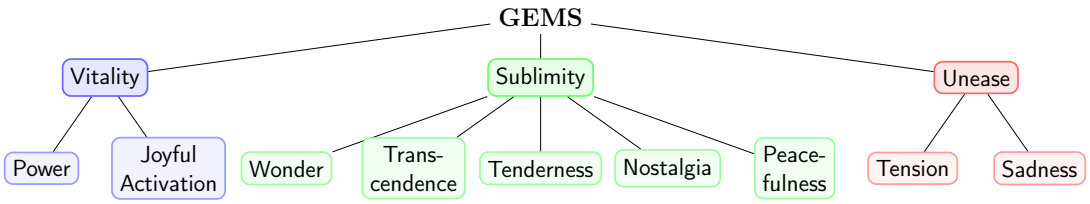


Figure 8.1.: Geneva Emotion Music Scale (GEMS) with 9 dimensions based on the factor analysis in [54].

for induced emotions as was later shown by neuroimaging work [45]. Consequently a MER approach based on this model can capitalize on a rich spectrum of music-specific emotional information [2].

As mentioned previously, the number of tracks in MER datasets is limited due to the scalability challenges associated with the annotation process. This limitation impacts the ability of supervised learning approaches to generalize effectively across a vast track catalog, as the availability of annotated data directly influences model performance. Semi-supervised learning on the other hand allows us to effectively incorporate information of unlabeled tracks as well as labeled ones in the learning process, leading to enriched track embeddings for the final labeling task. Moreover, prior works often ignore user and track meta-data which could be utilized to improve the learning process.

In this paper, we propose a novel framework employing a **S**emi-supervised Multi-**R**elational **G**raph **N**eural **N**etwork for **Emo**tion Recognition (SRGNN-Emo) for predicting the emotion profiles of tracks. We define the emotion profile of a music track as the set and intensity of emotions that the track evokes in listeners [25, 41]. Unlike traditional MER approaches, our model advances by adopting a multi-target regression strategy, aiming to capture more accurately the broad spectrum of emotions sparked through music. Building upon the premise that human listening behaviors encapsulate a wealth of information about evoked emotions, we innovate by integrating semi-supervised learning with human annotations and a multi-relational graph framework. This integration allows us to exploit the rich, albeit underutilized data from user interactions, genres, and tags, hypothesizing that such data, when structured into diverse graph formats and refined by a semi-supervised learning framework induces valuable emotion-related information. Our framework can predict emotional intensities across 9 dimensions, significantly enhancing the emotional insights derived from track embeddings compared to traditional methods that typically rely on fewer, music-nonspecific emotion dimensions such as valence and arousal.

Our approach not only aims to mitigate the limitations imposed by the scarcity of large, annotated datasets but also introduces a novel perspective on utilizing multi-relational graph structures to enrich track representations. To summarize, the main technical contributions of our work are as follows:

- We propose a novel multi-relational graph structure, based on user interactions, genres, and tags.
- We integrate a semi-supervised learning approach for multi-target regression into the framework of GNNs.
- We use a high-quality dataset based on state-of-the-art psychological research into music-evoked emotions for fine-grained MER [41].
- Extensive experiments show that our proposed model significantly outperforms state-of-the-art competitors on the task of MER.

To ensure reproducibility, we will release the code of our experiments and model weights on GitHub[1].

8.2. Related Work and Background

8.2.1. Music Emotion Recognition

MER aims to understand and categorize emotions in music through computational means. Key contributions to this field address the different facets of music and emotion, proposing various methodologies for recognition and analysis [50]. [25] present a comprehensive overview of MER, introducing a computational framework that generalizes emotion recognition from categorical domains to a 2D space defined by valence and arousal, facilitating novel emotion-based music retrieval and organization methods. Other works [12, 34, 49, 53] emphasize the role of integrating lyrics, chord sequences, and genre metadata alongside audio features, demonstrating how multifaceted approaches can significantly enhance MER systems' accuracy.

The development of MER has also been propelled by the creation of extensive datasets and embeddings tailored for this purpose. For instance, the MuSe dataset [1], which includes 90,000 tracks annotated with arousal, valence, and dominance values inferred from tags. Moreover, works such as those by [4, 7, 9] have evaluated various audio embeddings, including *Jukebox* and *musicnn* embeddings, for their effectiveness in MER

[1] https://github.com/dbis-uibk/SRGNN-Emo

tasks. Additionally, recent evaluations of state-of-the-art music audio embeddings have been conducted using tasks like the MediaEval challenge series on Emotion and Theme Recognition in Music [44] on the MTG-Jamendo mood/theme auto-tagging dataset [8].

Advances in MER research have also been characterized by the development of novel features and the design of sophisticated machine learning models. [6] show the effectiveness of using physiological signals, specifically EEG, to recognize emotions elicited by different music genres, highlighting the potential of brain signals in providing insights into emotional responses to music. [35] improve music emotion classification by introducing highly emotionally relevant audio features related with music performance expressive techniques or musical texture. The application of deep learning techniques has shown promising results in recognizing emotions from music, as seen in the work by [56]. They extracted features from log Mel-spectrograms by multiple parallel convolutional blocks and applied attention in combination with a sequence learning model for dynamic music emotion prediction. Others propose to structure musical features from different modalities (audio and lyrics) over a heterogeneous network to incorporate different modalities in a unique space for MER [13].

Our proposed approach SRGNN-Emo innovates by leveraging semi-supervised learning with user interaction data and metadata for nuanced emotional profiles, extending beyond traditional supervised methods.

8.2.2. Semi-Supervised Node Representation Learning

Node representation learning is focused on creating simplified vector representations of a graph's nodes that reflect both their connections and features. Traditional methods (without deep learning) are mostly based on random walks to examine the neighborhoods around nodes [17, 36, 42].

Graph Neural Networks (GNNs) are neural architectures specifically tailored for graph-structured data. GNNs learn meaningful node representations by iteratively aggregating and transforming information from a node's neighbors, effectively capturing complex relational and structural dependencies in graphs [18, 27]. Since the introduction of Graph Convolutional Networks (GCNs) [27, 47], a specific type of GNNs, more advanced techniques for node embedding have been developed, including a layer sampling algorithm [18] designed to work with large graphs by focusing on a set neighborhood of nodes.

Recently, we have observed a shift towards self-supervised contrastive approaches. These methods distinguish between positive (similar neighborhood) and negative (far away in the graph) examples to compute loss. Deep Graph Infomax (DGI) [48] enhances the mutual information between individual nodes and the whole graph representations. [19] introduce a method for learning representations from different viewpoints by contrasting nearby neighbor encodings with those from a more extensive graph diffusion. However,

because contrastive learning often requires a significant number of negative examples, it can be challenging to scale for large graphs. An alternative proposed by [43] named Bootstrapped Graph Latents (BGRL) avoids this issue by predicting alternative augmentations of the input, eliminating the need for contrasting with negative samples.

Despite significant advances in node representation learning, relatively little attention has been given to multi-relational graph neural networks and their application in specific domains like MER. Existing works such as [40], who proposed relational graph convolutional networks (R-GCNs) for knowledge graph completion, and [46], who explored compositional embeddings for relationships, have made strides in handling complex relational structures. However, these approaches have not been widely explored within the context of semi-supervised learning. Additionally, although semi-supervised node representation learning has become increasingly popular in tasks such as node classification and link prediction [18, 27], its application to emotion recognition tasks remains rare and under-investigated [20].

In this paper, we present an innovative framework that aligns with recent trends towards contrastive learning in GNNs but also extends them by specifically addressing the multi-relational and semi-supervised nature of the problem space in MER.

8.3. Dataset

In this work, we will leverage high-quality data from psychology-informed user studies on emotions evoked by music. We utilize the Emotion-to-Music Mapping Atlas (EMMA)[2] database [41], which comprises 817 music tracks. These tracks were annotated in 2023 based on their emotional impact, as assessed using GEMS [54]. We focus on the GEMS-9 variant of this scale, which is a checklist version of the original 45-item GEMS that assesses each dimension with one item only. Previous research has demonstrated emotion profiles derived from the original GEMS and the GEMS-9 to be highly correlated [23]. Emotions induced by each track were rated on these dimensions by an average of 28.76 annotators. We are one of the first to leverage this information-rich dataset for MER purposes, demonstrating the significant potential it offers for advancing research in this field. To enhance the reliability of our analyses, we restrict our focus to tracks with a higher interrater agreement, selecting those with an Intraclass Correlation Coefficient (ICC) above 0.5, which indicates moderate reliability [41]. While a higher ICC threshold would ensure even greater reliability, it would significantly reduce the dataset size, thereby limiting the diversity and generalizability of the data. However, it is worth mentioning that the ICC across all tracks demonstrates good interrater agreement, with a mean ICC value of 0.8.

[2] https://musemap-tools.uibk.ac.at/emma/

As our goal is to design a model for large-scale emotion recognition in a semi-supervised manner, we require a dataset containing rich information about the audio, but also relevant meta-data. Therefore, we employ the Music4All-Onion [32] dataset. This dataset enhances the Music4All [39] dataset by incorporating 26 additional audio, video, and metadata characteristics for 109,269 music pieces. It also includes 252,984,396 listening records from 119,140 Last.fm[3] users, enabling the use of user-item interactions. Intersecting EMMA with the Music4All-Onion dataset leads to 509 tracks with available emotion profiles, audio features, and meta information. Due to our hypothesis that human listening behavior in combination with track metadata encapsulates valuable information about evoked emotions, we extract graph structures from user listening sessions, track genres, and user tags as will be described in detail in Section 8.4.1.

For each track available in the Music4All-Onion dataset we use pre-trained instances of *musicnn* [37], *MAEST* [4] and *Jukebox* [14] to represent the audio signal. The *musicnn* model is based on deep convolutional neural networks trained to classify music based on its content [37]. The *MAEST* representations are based on spectrogram-based audio transformers which employ patchout training on a supervised task [4]. *Jukebox* is a generative model for music that uses a deep neural network trained on a vast corpus of tracks to understand and generate music [14].[4] These models were selected for their music-specific design, which ensures a closer alignment with musical features like melody, harmony, and rhythm that are critical for emotion recognition [31].[5]

8.4. Proposed Method (SRGNN-Emo)

In this section, we introduce a novel framework leveraging a multi-relational graph structure and semi-supervised learning. Multi-relational graphs are complex data structures that model different types of relations that correspond to different user data types in our case (e. g., Figure 8.2). Our model is designed to extract emotional profiles from music tracks by integrating rich user interaction data with diverse metadata and sophisticated content data. Figure 8.2 provides an overview of our proposed approach, where each module will be explained in the following.

[3] https://www.last.fm

[4] For *MAEST*, embeddings were extracted from transformer block 7 of the model, initialized with PaSST weights and pre-trained on the Discogs20 dataset. For *Jukebox*, embeddings were extracted from layer 36, with mean pooling applied across the layer's output, following the methodology detailed in the original work.

[5] This extended version of the dataset, including audio embeddings extracted from the described pre-trained models (*musicnn*, *MAEST*, and *Jukebox*), is made publicly available on https://zenodo.org/records/15394646.

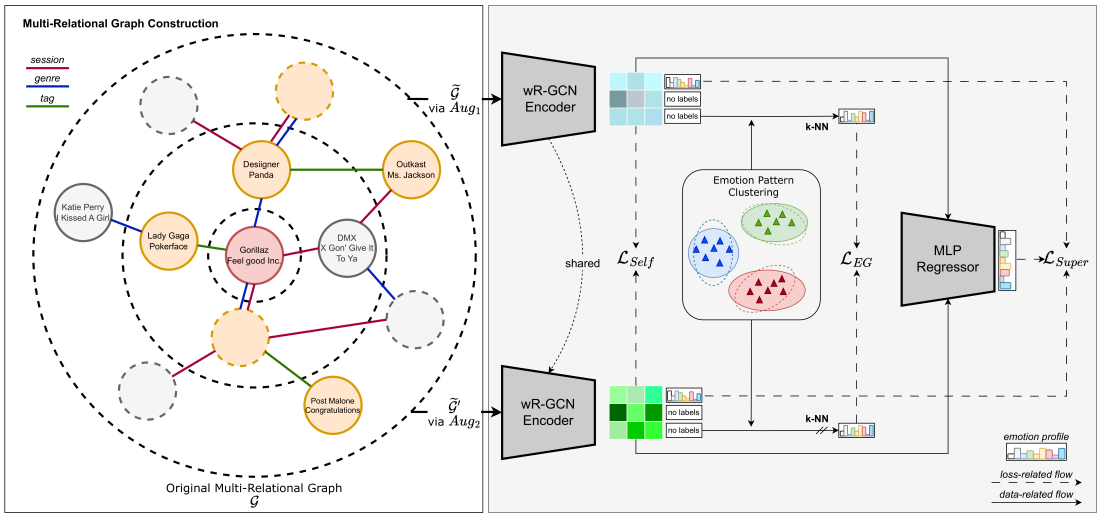


Figure 8.2.: Illustration of SRGNN-Emo which constructs a multi-relational graph with nodes representing tracks, and edges symbolizing connections based on sessions, genres, or user tags shared among tracks. We use stochastic graph augmentations to generate two distinct graph views, which are processed by a shared encoder to ensure robust and invariant node representations in a self-supervised manner. The emotion-guided consistency objective ($\mathcal{L}_{EG}$) optimization aims to align unlabeled nodes with emotion profile patterns of labeled nodes across augmented graph views. The learned node representations are then fed into a Multi-Layer Perceptron (MLP) Regressor to predict the emotion profile of each track.

8.4.1. Multi-Relational Graph Construction

We aim to derive representations of tracks that encapsulate nuanced similarities between music tracks, based on shared genres, commonality in listening sessions, and user-assigned tags. Therefore, we construct a multi-relational graph G, focusing on tracks as nodes, with edges representing different types of relationships such as sessions, genres, or tags that connect these tracks. Specifically, nodes in our multi-relational graph are tracks $v \in V$, and an edge $(v_i, v_j) \in E$ is established between two tracks if they are part of the same listening session by a user, share one or multiple genres, or have been tagged with one or multiple identical tags by users. The strength of the connection, represented as the edge weight $e_{ij}^{(r)}$, reflects the frequency of shared relationships $r \in R$, such as the number of common tags, genres, or sessions. We normalize the edge weights per relation such that, for each track v and each relation r, the edge weights are symmetrically scaled using the formula:

$$\tilde{e}_{ij}^{(r)} = \frac{e_{ij}^{(r)}}{\sqrt{\deg(v_i) \cdot \deg(v_j)}}$$

where $\deg(v)$ represents the degree of node v for relation r. This symmetric normalization ensures that, for each track v and each relation r, the edge weights are adjusted based on the degrees of both connected nodes and therefore mitigates the inherent popularity bias of tracks.

8.4.2. Emotion-Based Graph Encoder

To learn node representations on this multi-relational graph G introduced before, we employ a weighted Relational Graph Convolutional Network (wR-GCN) encoder, which adapts the GNN message-passing framework to handle the complexities of a multi-relational graph [40] and additionally incorporates edge weights. The GNN message-passing framework [15] enables nodes to exchange and integrate information with their neighbors, iteratively refining their representations to capture the graph's structural and relational context. The general differentiable message passing is formulated as:

$$h_i^{(l+1)} = \sigma\left(\sum_{m \in \mathcal{M}_i} g_m\left(h_i^{(l)}, h_j^{(l)}\right)\right), \tag{8.1}$$

where $h_i^{(l)} \in \mathbb{R}^{d^{(l)}}$ represents the hidden state of node v_i at the l-th layer, with $d^{(l)}$ being the dimensionality of the layer's representation. The incoming messages, $g_m(\cdot,\cdot)$, are combined and processed through an activation function $\sigma(\cdot)$, such as ReLU. $\mathcal{M}_i$ is the set of incoming messages for node v_i, typically corresponding to the set of incoming edges. The function $g_m(\cdot,\cdot)$ is often a neural network or a simple linear transformation [27].

This transformation has proven effective in accumulating and encoding features from local, structured neighborhoods [27, 47]. For our multi-relational, weighted graph we define a simple propagation model [40] for computing the forward-pass update of a node v_i and extend it with the usage of edge weights:

$$h_i^{(l+1)} = \sigma\left(\sum_{r \in R} \sum_{j \in \mathcal{N}_i^r} \frac{1}{|\mathcal{N}_i^r|} W_r^{(l)} h_j^{(l)} e_{ij}^{(r)} + W_0^{(l)} h_i^{(l)} e_{ii}^{(r)}\right), \tag{8.2}$$

where $\mathcal{N}_i^r$ represents the set of neighbors of node v_i under relation $r \in R$, and $e_{ij}^{(r)}$ is the edge weight between nodes v_i and v_j for relation r. This equation intuitively accumulates the transformed feature vectors of neighboring nodes through a weighted and normalized sum. Unlike regular GCNs, we incorporate relation-specific transformations, depending on the type and direction of the edge. Additionally, to ensure that the node's representation at layer $l+1$ is informed by its representation at layer l, we introduce a self-connection under each relation type for each node.

Initially, $h_v^0 = x_v$, representing the node features. We use the corresponding representations of the tracks (e. g., *musicnn*, *MAEST* or *Jukebox*) as the node features $X \in \mathbb{R}^{N \times F}$,

where N is the number of nodes in the graph and F the feature dimension. We define $\mathcal{N}(v, r)$ as a uniformly sampled neighborhood across all relations $r \in R$ to manage memory and computation effectively [18].

8.4.3. Semi-Supervised Multi-Target Regression

Contrastive learning has shown to be a valuable paradigm for self-supervised learning and consistency regulation in the context of GNNs [29, 43]. We employ this idea as the grounding learning task for our graph-based model and extend it with a semi-supervised loss in the process.

Given an input graph, we generate two distinct graph views through stochastic graph augmentations. These augmentations involve randomly masking different node features and dropping a different subset of edges per graph to introduce variability. The resulting augmented graph views are denoted by $\widetilde{G} = (\widetilde{A}, \widetilde{X})$ and $\widetilde{G}' = (\widetilde{A}', \widetilde{X}')$, where $\widetilde{A}$ and $\widetilde{A}'$ represent the adjacency matrices of the augmented graphs, and $\widetilde{X}$ and $\widetilde{X}'$ denote the feature matrices post-augmentation.

Representation Learning via Shared Encoder

To learn robust, low-dimensional node-level representations we employ a shared encoder strategy that learns consistent representations across different graph augmentations. Both augmented graph views are input into our shared wR-GCN encoder, denoted as $f_\theta : \mathbb{R}^{N\times N} \times \mathbb{R}^{N\times F} \to \mathbb{R}^{N\times D}$, to learn low-dimensional node-level representations. The node-level representations obtained from the encoder for the two views are $f_\theta(\widetilde{A}, \widetilde{X}) = \widetilde{Z} \in \mathbb{R}^{N\times D}$ and $f_\theta(\widetilde{A}', \widetilde{X}') = \widetilde{Z}' \in \mathbb{R}^{N\times D}$, respectively.

To ensure the learned node representations are invariant to the augmentations, SRGNN-Emo minimizes the cosine distance between the representations from the two differently augmented views on a node-wise basis and is formalized as follows:

$$\mathcal{L}_{\text{Self}} = \frac{1}{N} \sum_{i=1}^{N} \frac{\widetilde{Z}_i \cdot \widetilde{Z}'_i}{\|\widetilde{Z}_i\| \|\widetilde{Z}'_i\|} \tag{8.3}$$

In their experiments, [29] found that using a single shared encoder in combination with subsequent supervisory signals was sufficient to prevent representation collapse, while also offering the benefits of simplicity and efficiency.

Emotion-Guided Consistency Objective

While our framework effectively leverages self-supervised learning signals—patterns and features extracted from unlabeled data without explicit supervision—through contrastive learning, it has yet to incorporate the limited but accessible emotion profiles available

for tracks. To leverage emotion label information effectively, we refine our method by aligning them with emotion profile patterns. Starting with a set of labeled tracks with known emotion profiles, we identify distinct emotion patterns through clustering, which then serve as reference points (centroids) in the emotion profile space. Our goal is to group the unlabeled tracks around these centroids, ensuring their predicted emotion profiles remain consistent across differently augmented views of the graph. By doing so, we aim to maximize the consistency and reliability of node assignments to these emotion patterns, effectively bridging the gap between labeled and unlabeled tracks.

Given the set of labeled tracks, denoted as V_L, we apply a k-means clustering algorithm to extract K distinct clusters, each representing a unique emotion pattern. The result is a set of centroids $C = \{c_1, c_2, \ldots, c_K\}$, where each $c_k \in \mathbb{R}^{1\times 9}$ corresponds to the centroid of cluster k. These nine dimensions correspond to the emotional dimensions defined by GEMS, which serve as the basis for clustering. For each unlabeled track v_{ul}, we compute the predicted emotion profile using a non-parametric weighted k-nearest neighbors (k-NN) approach to generate pseudo-labels, formulated as:

$$p_i = \frac{\sum_{j\in \mathrm{NN}_k(H_i)} \mathrm{sim}(H_i, H_j^S)\cdot Y_j^S}{\sum_{j\in \mathrm{NN}_k(H_i)} \mathrm{sim}(H_i, H_j^S)} \tag{8.4}$$

where $\mathrm{sim}(\cdot,\cdot)$ computes the cosine similarity between two vectors, $H^S \in \mathbb{R}^{N\times D}$ and $Y^S \in \mathbb{R}^{L\times 9}$ denote the support (labeled) node representations and the emotion profiles, respectively and $\mathrm{NN}_k(H_i)$ denotes the set of $K_{\text{neighbors}}$ nearest neighbors of H_i in H^S.

To enhance reliability, we restrict the k-NN predictions to only confident pseudo-labels by measuring the distance between each pseudo-label and the centroids C. We retain nodes whose predicted profile shows a similarity above a threshold μ with at least one centroid, forming the set V_{conf}. The emotion-guided consistency objective is then defined as:

$$\mathcal{L}_{EG} = \frac{1}{|V_{conf}|} \sum_{v_i\in V_{conf}} \mathrm{MSE}(\tilde{\mathbf{p}}_i, \tilde{\mathbf{p}}_i'), \tag{8.5}$$

where $\mathrm{MSE}(\cdot,\cdot)$ denotes the mean squared error loss function and $\tilde{\mathbf{p}}_i$ and $\tilde{\mathbf{p}}_i'$ are the confident predicted emotion profiles for track v_i from the augmented graphs. Using a high value for μ prioritizes confident pseudo-labels in the objective function, which has been shown to effectively mitigate confirmation bias [5, 29].

This approach not only incorporates label information to guide the learning of emotion profile patterns but also ensures that predictions for unlabeled tracks are made with higher confidence, thereby improving the overall model's ability to generalize from labeled to unlabeled data in the context of a multi-target regression task.

Emotion Profile Prediction

After learning robust node representations through the shared wR-GCN encoder and ensuring consistency across augmented graph views, the final step of SRGNN-Emo is to predict the emotion profile for each music track. To achieve this, we utilize a Multi-Layer Perceptron (MLP) $\mathcal{R}(\cdot)$ that takes as input the averaged node representations from the two augmented views and outputs the emotion profile per node/track. The MLP consists of three fully connected layers, each followed by a LeakyReLU activation function and a dropout layer to prevent overfitting. The output of the MLP is a vector $\hat{\mathbf{y}}_i \in \mathbb{R}^9$, representing the predicted emotion intensities across the nine emotion categories. To train the model to predict nine continuous emotion dimensions, we employ a mean squared error (MSE) loss as our supervised objective:

$$\mathcal{L}_{\text{Super}} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{9} \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2 \tag{8.6}$$

8.4.4. Final Objective

The combined objective function for SRGNN-Emo is expressed as:

$$\mathcal{L} = \alpha \mathcal{L}_{\text{Self}} + \beta \mathcal{L}_{\text{EG}} + \mathcal{L}_{\text{Super}} \tag{8.7}$$

where α and β are coefficients that control the contribution of the self-supervised loss $\mathcal{L}_{\text{Self}}$ and the emotion-guided loss $\mathcal{L}_{\text{EG}}$ to the overall training objective, respectively. The supervised loss $\mathcal{L}_{\text{Super}}$ ensures the model effectively predicts continuous emotion profiles for labeled nodes.

8.5. Experiments and Results

We compare SRGNN-Emo against traditional and graph-based baselines for MER. In the following, we detail our experimental setup, including data preparation, model configurations, and metrics used for evaluation.

8.5.1. Baselines

We systematically compare our proposed model against a diverse array of baseline approaches, spanning traditional machine learning models, graph-based approaches, and a novel custom convolutional neural network, each harnessing unique feature representations from music analysis frameworks.

We begin with traditional machine learning models including Logistic Regression (LR) and Support Vector Regression (SVR). Additionally, Co-training Regression (COREG) [57]

is utilized, enhancing generalization by co-training two regressors on separate views. A Multilayer Perceptron (MLP) model with three layers (the same as in our SRGNN-Emo model) serves a dual purpose: it depicts a baseline on its own and acts as the regressor for semi-supervised learning task in the graph-based models (with the learned node representations as input).

The graph-based models are crucial for understanding the relational structure of music data. This category includes Label Propagation (LP) [58] for emphasizing data clustering, Graph Convolutional Network (GCN) [27] and Graph Attention Network (GAT) [47] for integrating node features with the graph topology, Deep Graph Infomax (DGI) [48] focusing on mutual information maximization, and Bootstrapped Graph Latent Representation (BGRL) [43] aimed at enhancing robustness through consistent node representation across views. MRLGCN [13] structures musical features over a heterogeneous network and learns a multi-modal representation using a GNN with features extracted from audio and lyrics for MER.

Completing our set of baselines, we designed a fully supervised, content-based, end-to-end method named DOMR+ for **D**ensity-based **O**versampling for **M**ultivariate **R**egression with data transformation. The DOMR+ method consists of two components: a Fully Convolutional Network model and a pre-processing stage. The model employs multiple convolutional and subsampling layers without dense layers. To address the challenges of data scarcity and imbalance in the labels, the pre-processing stage integrates oversampling with data transformation techniques. Candidate data points for oversampling are identified using kernel density estimation (KDE), which determines the rarity of data points based on their density within the feature space. Instead of directly oversampling these candidates, the method applies class-preserving audio transformations, which minimally transforms the original audio while retaining its fundamental properties, including filtering, equalizing, noise addition, scale changes (pitch shifting and time stretching), distortions, quantization, dynamic compression, format encoding/decoding (e. g., MP3, GSM) and reverberation [30]. These transformations ensure that the augmented data remain representative of the underlying distribution, enhancing the model's ability to generalize, while avoiding the risk of overfitting caused by repetitive synthetic samples.

8.5.2. Experimental Setup

We preprocessed the target variables representing emotions by applying z-normalization, which ensures each variable has a mean of 0 and a standard deviation of 1. We employed stratified 10-fold cross-validation based on binning to validate the performance of our models comprehensively.

For performance evaluation, we rely on two metrics: Root Mean Square Error (RMSE) and coefficient of determination (R^2). RMSE measures the average magnitude of the errors between the predicted and actual values. A lower RMSE indicates better perfor-

Model \ Rep.	musicnn		MAEST		Jukebox	
	RMSE↓ (±SE)	R^2↑ (±SE)	RMSE↓ (±SE)	R^2↑ (±SE)	RMSE↓ (±SE)	R^2↑ (±SE)
LR	0.8443 (±0.02)	0.2470 (±0.05)	1.3821 (±0.06)	-1.0731 (±0.22)	1.0301 (±0.04)	-0.1403 (±0.09)
SVR	0.8188 (±0.01)	0.2968 (±0.01)	0.7862 (±0.01)	0.3504 (±0.02)	0.9802 (±0.02)	0.0163 (±0.01)
COREG	0.8742 (±0.02)	0.1140 (±0.05)	0.8613 (±0.02)	0.1346 (±0.08)	0.8680 (±0.02)	0.1244 (±0.05)
MLP	0.8132 (±0.02)	0.3106 (±0.02)	0.8938 (±0.03)	0.1576 (±0.08)	0.8579 (±0.02)	0.2193 (±0.06)
LP†	0.9488 (±0.03)	0.0806 (±0.01)	0.9488 (±0.03)	0.0806 (±0.01)	0.9488 (±0.03)	0.0806 (±0.01)
GCN	0.8071 (±0.02)	0.3158 (±0.04)	0.7781 (±0.02)	0.3568 (±0.05)	0.7492 (±0.04)	0.4039 (±0.05)
GAT	0.8167 (±0.03)	0.2992 (±0.07)	0.7856 (±0.02)	0.3476 (±0.05)	0.7567 (±0.02)	0.3926 (±0.03)
DGI	0.8042 (±0.02)	0.3184 (±0.06)	<u>0.7749</u> (±0.01)	0.3644 (±0.06)	<u>0.7464</u> (±0.02)	<u>0.4103</u> (±0.04)
BGRL	<u>0.8019</u> (±0.02)	<u>0.3253</u> (±0.05)	0.7939 (±0.02)	0.3370 (±0.07)	0.7905 (±0.02)	0.3843 (±0.05)
MRLGCN	0.8592 (±0.04)	0.2600 (±0.04)	0.7868 (±0.03)	<u>0.3648</u> (±0.05)	0.7932 (±0.03)	0.3651 (±0.05)
DOMR+†	0.8291 (±0.03)	0.2777 (±0.08)	0.8291 (±0.03)	0.2777 (±0.08)	0.8291 (±0.03)	0.2777 (±0.08)
SRGNN-Emo	**0.7973** (±0.03)	**0.3305** (±0.06)	**0.7707** (±0.01)	**0.3724** (±0.05)	**0.7411** (±0.02)	**0.4180** (±0.04)

Table 8.1.: Multi-target regression performance for different models across three representation types. The best results are in boldface and the second-best results are underlined. All improvements of SRGNN-Emo compared to the second-best performing model are significant (Wilcoxon signed-rank test, $p < .05$). Models marked with † do not use any underlying track representation.

mance. R^2, on the other hand, is a goodness-of-fit measure for regression models and assesses the proportion of variance in the dependent variable that is predictable from the independent variables, with values closer to 1 indicating better model fit.

All baseline models are carefully tuned via grid search, optimizing hyperparameters including (but not limited to) number of layers $\in \{1, \ldots, 5\}$, number of neighbors $\in \{5, 10, \ldots, 50\}$, learning rate, dropout and regularization strength, depending on the respective model requirements. For our proposed model, SRGNN-Emo, the Adam optimizer [26] is used, with the learning rate set to 0.001 and L_2 regularization set to 10^{-5}. We tuned its hyperparameters within specific ranges: the number of layers L in the wR-GCN was set between 1 to 5, the number of neighbors was chosen from between 5 and 50, and the α and β weight parameters were logarithmically adjusted within the range of 0.1 to 10. Additionally, dropout rates were varied between 0.0 and 0.5 to prevent overfitting. The number of clusters K and nearest-neighbors $K_{\text{neighbors}}$ is searched in $\{2, 4, 6, \ldots, 16\}$ and $\{5, 10, 20, 40\}$, correspondingly.

8.5.3. Performance Analysis

Table 8.1 summarizes the multi-target regression performance of various models, including traditional machine learning methods, graph-based models, and our proposed SRGNN-Emo framework. The results demonstrate that SRGNN-Emo achieves the lowest RMSE and highest R^2 score, indicating superior prediction performance (statistically significant) and model fit, respectively.

Model	wond	tran	tend	nost	peace	joya	power	sadn	tens	GEMS-9
MLP (musicnn)	0.9312	0.9653	0.7330	0.8936	0.6466	0.8099	0.8007	0.7711	0.7675	0.8132
DGI (Jukebox)	0.9059	0.9425	0.6647	0.8094	<u>0.6088</u>	0.7162	0.7511	**0.6627**	**0.6569**	0.7464
SRGNN-Emo (Jukebox)	**0.8972**	**0.9345**	**0.6518**	**0.8026**	0.6162	**0.6930**	**0.7425**	<u>0.6690</u>	<u>0.6630</u>	**0.7411**
(A) w/o $\mathcal{L}_{\text{Self}}$	0.9177	0.9384	0.6532	0.8192	**0.6086**	0.7050	0.7653	0.6713	0.6829	0.7513
(B) w/o $\mathcal{L}_{\text{EG}}$	0.9041	0.9387	0.6636	0.8245	0.6110	0.7082	0.7650	0.6845	0.6779	0.7530
(C) w/o $\mathcal{L}_{\text{Super}}$	1.2372	1.0996	1.2454	1.2210	1.3543	1.3329	1.2424	1.2907	1.2339	1.2508

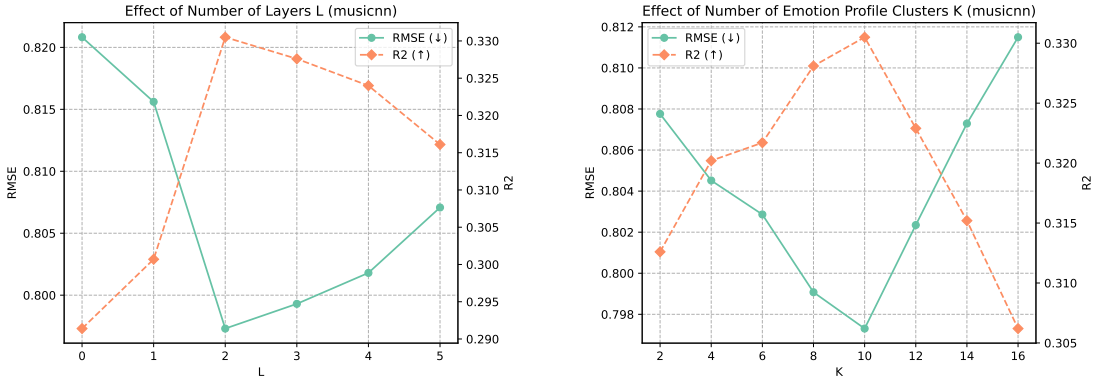
Table 8.2.: RMSE scores of models (using the best-performing representations from Table 8.1) across multiple emotion targets. Abbreviations of emotion dimensions correspond to Wonder, Transcendence, Tenderness, Nostalgia, Peacefulness, Joyful Activation, Power, Sadness, and Tension. All improvements of the best-performing models (boldface) are statistically significant compared to the second-best models (underline) per emotion dimension (Wilcoxon signed-rank test, $p < .05$).

Representing traditional machine learning approaches LR, SVR, and COREG show relatively higher RMSE values, indicating lower predictive performance. Their R^2 values are also significantly lower, confirming less variance explained by these models. The baseline MLP shows competitive performance when relying on *musicnn* representations, but is outperformed by graph-based approaches with the other two representations (*MAEST* and *Jukebox*).

Among the graph-based approaches, DGI and BGRL show competitive performance with the lowest RMSE and highest R^2 among the graph-based models for two different representations, ranked second after our SRGNN-Emo. GCN and GAT also demonstrate robust performances but are slightly outperformed by DGI or BGRL, depending on the underlying representation. Our model SRGNN-Emo outperforms all baseline models and indicates a statistically significant improvement in terms of RMSE and R^2 compared to the second-best models, DGI and BGRL.

8.5.4. Ablation Study

The ablation study, detailed in Table 8.2, assesses the impact of individual components of SRGNN-Emo by removing $\mathcal{L}_{\text{Self}}$, $\mathcal{L}_{\text{EG}}$, and $\mathcal{L}_{\text{Super}}$ separately. The results illustrate the essential roles of these components in the model's overall performance. Removing the self-supervised loss ($\mathcal{L}_{\text{Self}}$) slightly increases the RMSE across 5 out of 9 emotional dimensions, suggesting that this component helps in stabilizing the learning process by enforcing consistent node representations across different graph augmentations. The removal of the emotion-guided consistency objective ($\mathcal{L}_{\text{EG}}$) leads to a noticeable degradation in performance across 8 out of 9 emotional dimensions. This confirms that $\mathcal{L}_{\text{EG}}$ plays a crucial role in refining node embeddings by aligning them more closely with known emotion profile patterns, thus enhancing the model's ability to generalize from labeled to unlabeled data. Omitting the supervised loss ($\mathcal{L}_{\text{Super}}$) results in significant performance drops across all emotional dimensions, with RMSE scores rising substantially.



(a) Performance impact of different number of layers L in our wR-GCN component.

(b) Performance impact of different number of emotion profile clusters K.

Figure 8.3.: Impact of hyperparameters on model performance using *musicnn* representations.

This drastic decline highlights the importance of direct supervision in guiding the network towards accurate emotion profile predictions. Interestingly, while DGI outperforms SRGNN-Emo in two emotional dimensions—Sadness and Tension—it does not achieve consistently better performance across all emotion dimensions, indicating limitations in its ability to fully capture the emotional variations present in the dataset.

8.5.5. Impact of Hyper-Parameters

In this section, we investigate the impact of different hyper-parameters. We focus on the number of layers L in the wR-GCN and the number of emotion profile clusters K, since these hyper-parameters are related to various parts of the model architecture. Figure 8.3a shows the performance of our model with different settings of layers L on the described dataset using *musicnn* representations. A higher number of layers in the multi-relational network does not necessarily lead to an increase in performance due to the issue of over-smoothing, where node representations converge to the same values [11, 27]. For our dataset, we can find a sweet spot layer setting L of 2.

Figure 8.3b shows the performance differences between runs relying on *musicnn* representations with a different number of emotion profile clusters. The best-performing setting for K is 10 clusters, which aligns with previous analyses of emotion profiles in GEMS-9 by [10].

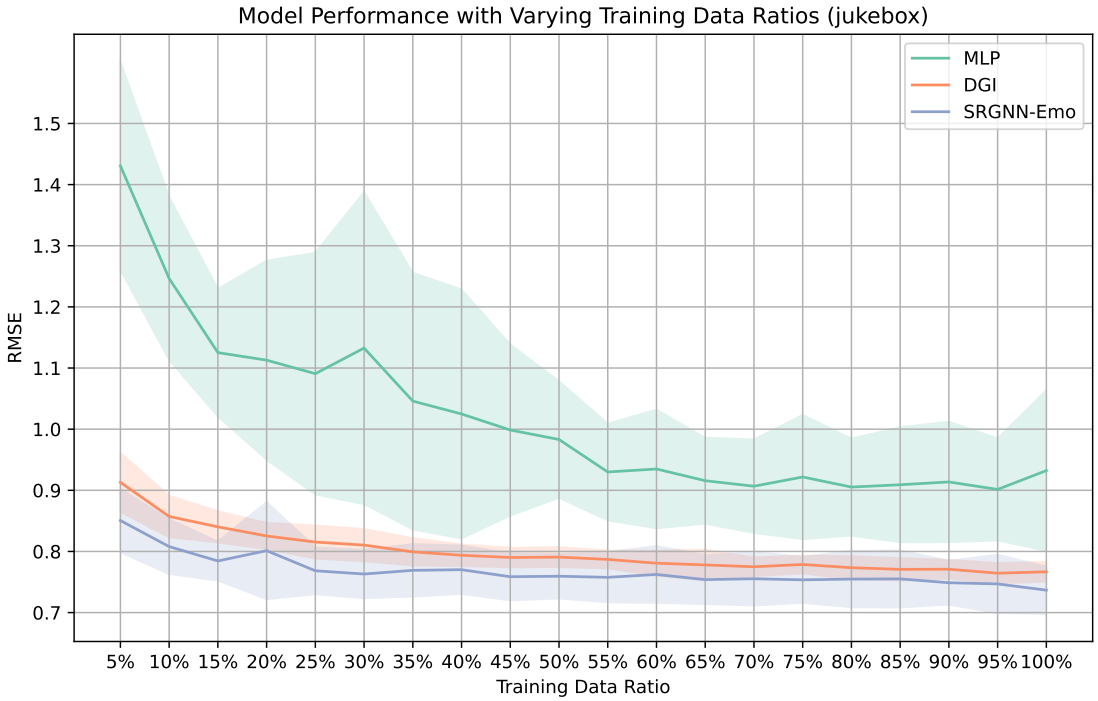


Figure 8.4.: Model performances on different fractions of training data using *Jukebox* representations.

8.5.6. Data Efficacy Study

In this section, we assess the efficacy of our proposed SRGNN-Emo framework under varying levels of training data availability, investigating its performance in semi-supervised settings where labeled data is sparse. Figure 8.4 illustrates the model performances using different ratios of the training data, comparing the SRGNN-Emo framework with baseline models, including the baseline MLP and the semi-supervised graph-based approach DGI.

The results show that as the amount of available labeled data increases, the performance of the MLP model significantly improves, exhibiting lower RMSE and higher R^2 values. This highlights its heavy reliance on large amounts of labeled data for generalization. In contrast, the semi-supervised models demonstrate superior performance even with minimal labeled data. Specifically, our SRGNN-Emo maintains competitive RMSE scores and high R^2 values across various fractions of the training data, showing only a gradual decline in prediction accuracy as the training set size reduces. This indicates the model's robustness in scenarios with limited labeled data.

8.6. Conclusion

This work introduced SRGNN-Emo, a novel Semi-supervised Multi-relational Graph Neural Network designed for nuanced MER trained on EMMA, a database with exceptionally rigorous annotations based on the domain-specific GEMS emotion model. By integrating semi-supervised learning with multi-relational graph structures and leveraging rich user interaction data, SRGNN-Emo effectively outperforms baseline models in capturing the complex emotional responses evoked by music. While our study leverages the GEMS model to capture a wide range of music-evoked emotions, our framework remains inherently flexible and can be adapted to alternative emotion models as future work. As a contribution, we enrich the existing Music4All-Onion dataset [32] by adding emotion labels generated from our trained model, resulting in a fully labeled large-scale emotion-based dataset with 109,269 tracks. This enhanced dataset enables various applications such as improved music retrieval, enhanced recommendation systems, and other related tasks.

References

[1] C. Akiki and M. Burghardt. Muse: the musical sentiment dataset. *Journal of Open Humanities Data*, 7, July 2021. DOI: 10.5334/johd.33.

[2] A. Aljanaki, F. Wiering, and R. C. Veltkamp. Computational modeling of induced emotion using GEMS. In *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014*, pages 373–378, 2014.

[3] A. Aljanaki, Y.-H. Yang, and M. Soleymani. Developing a benchmark for emotional analysis of music. *PloS one*, 12(3):e0173392, 2017.

[4] P. Alonso-Jiménez, X. Serra, and D. Bogdanov. Efficient supervised training of audio transformers for music representation learning. In *Proceedings of the 24th International Society for Music Information Retrieval Conference, ISMIR 2023*, pages 824–831, 2023.

[5] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International joint conference on neural networks (IJCNN)*, pages 1–8. IEEE, 2020.

[6] A. M. Bhatti, M. Majid, S. M. Anwar, and B. Khan. Human emotion recognition and analysis in response to audio music using brain signals. *Computers in Human Behavior*, 65:267–275, 2016.

[7] D. Bogdanov, X. Lizarraga-Seijas, P. Alonso-Jiménez, and X. Serra. Musav: a dataset of relative arousal-valence annotations for validation of audio models. In *International Society for Music Information Retrieval Conference (ISMIR 2022)*, 2022.

[8] D. Bogdanov, M. Won, P. Tovstogan, A. Porter, and X. Serra. The mtg-jamendo dataset for automatic music tagging. In *Machine Learning for Music Discovery Workshop, International Conference on Machine Learning (ICML 2019)*, 2019.

[9] R. Castellon, C. Donahue, and P. Liang. Codified audio language modeling learns useful representations for music information retrieval. In *Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021*, pages 88–96, 2021.

[10] M. Chełkowska-Zacharewicz and M. Janowski. Polish adaptation of the geneva emotional music scale: factor structure and reliability. *Psychology of Music*, 49(5):1117–1131, 2021.

[11] T. Chen and R. C. Wong. Handling information loss of graph neural networks for session-based recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1172–1180. ACM, 2020.

[12] J. Choi, J.-H. Song, and Y. Kim. An analysis of music lyrics by measuring the distance of emotion and sentiment. In *2018 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pages 176–181. IEEE, 2018.

[13] A. C. M. da Silva, D. F. Silva, and R. M. Marcacini. Heterogeneous graph neural network for music emotion recognition. In *Proceedings of the 23rd International Society for Music Information Retrieval Conference, ISMIR 2022*, pages 667–674, 2022.

[14] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever. Jukebox: A generative model for music. *CoRR*, abs/2005.00341, 2020. arXiv: 2005.00341.

[15] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[16] J. S. Gómez-Cañón, E. Cano, T. Eerola, P. Herrera, X. Hu, Y.-H. Yang, and E. Gómez. Music emotion recognition: toward new, robust standards in personalized and context-sensitive applications. *IEEE Signal Processing Magazine*, 38(6):106–114, 2021.

[17] A. Grover and J. Leskovec. Node2vec: scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 2016*, pages 855–864. ACM, 2016.

[18] W. L. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 1024–1034, 2017.

[19] K. Hassani and A. H. K. Ahmadi. Contrastive multi-view representation learning on graphs. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, volume 119 of *Proceedings of Machine Learning Research*, pages 4116–4126. PMLR, 2020.

[20] A. Horner, D. H. Hu, B. Wu, Q. Yang, and E. Zhong. SMART: semi-supervised music emotion recognition with social tagging. In *Proceedings of the 13th SIAM International Conference on Data Mining*, pages 279–287. SIAM, 2013.

[21] X. Hu and J. S. Downie. When lyrics outperform audio for music mood classification: a feature analysis. In *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010*, pages 619–624. International Society for Music Information Retrieval, 2010.

[22] X. Hu, J. S. Downie, and A. F. Ehmann. Lyric text mining in music mood classification. *American music*, 183(5,049):2–209, 2009.

[23] P.-O. Jacobsen, H. Strauss, J. Vigl, E. Zangerle, and M. Zentner. Assessing aesthetic music-evoked emotions in a minute or less: a comparison of the gems-45 and the gems-9. *Musicae Scientiae*, 0(0):10298649241256252, 2024. DOI: 10.1177/10298649241256252.

[24] Z. Jia, Y. Lin, J. Wang, Z. Feng, X. Xie, and C. Chen. Hetemotionnet: two-stream heterogeneous graph recurrent neural network for multi-modal emotion recognition. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 1047–1056, 2021.

[25] Y. E. Kim, E. M. Schmidt, R. Migneco, B. G. Morton, P. Richardson, J. Scott, J. A. Speck, and D. Turnbull. Music emotion recognition: a state of the art review. In *Proc. ismir*, volume 86, pages 937–952, 2010.

[26] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*, 2015.

[27] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.

[28] C. Laurier, M. Sordo, J. Serra, and P. Herrera. Music mood representations from social tags. In *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009*, pages 381–386. International Society for Music Information Retrieval, 2009.

[29] J. Lee, Y. Oh, Y. In, N. Lee, D. Hyun, and C. Park. Grafn: semi-supervised node classification on graph with few labels via non-parametric distribution assignment. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2243–2248. ACM, 2022.

[30] R. Mignot and G. Peeters. An analysis of the effect of data augmentation methods: experiments for a musical genre classification task. *Trans. Int. Soc. Music. Inf. Retr.*, 2(1):97–110, 2019.

[31] M. Moscati, E. Parada-Cabaleiro, Y. Deldjoo, E. Zangerle, and M. Schedl. Music4all-onion. In Zenodo, May 2025. DOI: 10.5281/zenodo.15394646.

[32] M. Moscati, E. Parada-Cabaleiro, Y. Deldjoo, E. Zangerle, and M. Schedl. Music4all-onion - A large-scale multi-faceted content-centric music recommendation dataset. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4339–4343. ACM, 2022.

[33] M. Moscati, H. Strauß, P. Jacobsen, A. Peintner, E. Zangerle, M. Zentner, and M. Schedl. Emotion-based music recommendation from quality annotations and large-scale user-generated tags. In *Proceedings of the 32nd ACM Conference on User Modeling, Adaptation and Personalization, UMAP 2024*, pages 159–164. ACM, 2024.

[34] R. Panda, R. Malheiro, and R. P. Paiva. Audio features for music emotion recognition: a survey. *IEEE Transactions on Affective Computing*, 14(1):68–88, 2020.

[35] R. Panda, R. Malheiro, and R. P. Paiva. Novel audio features for music emotion recognition. *IEEE Transactions on Affective Computing*, 11(4):614–626, 2018.

[36] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: online learning of social representations. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14*, pages 701–710. ACM, 2014.

[37] J. Pons and X. Serra. Musicnn: pre-trained convolutional neural networks for music audio tagging. *CoRR*, abs/1909.06654, 2019. arXiv: 1909.06654.

[38] R. Rajan, J. Antony, R. A. Joseph, J. M. Thomas, et al. Audio-mood classification using acoustic-textual feature fusion. In *2021 Fourth International Conference on Microelectronics, Signals & Systems (ICMSS)*, pages 1–6. IEEE, 2021.

[39] I. A. P. Santana, F. Pinhelli, J. Donini, L. G. Catharin, R. B. Mangolin, Y. M. e Gomes da Costa, V. D. Feltrim, and M. A. Domingues. Music4all: A new music database and its applications. In *2020 International Conference on Systems, Signals and Image Processing, IWSSIP 2020*, pages 399–404. IEEE, IEEE, 2020.

[40] M. S. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer, 2018.

[41] H. Strauss, J. Vigl, P.-O. Jacobsen, M. Bayer, F. Talamini, W. Vigl, E. Zangerle, and M. Zentner. The emotion-to-music mapping atlas (emma): a systematically organized online database of emotionally evocative music excerpts. *Behavior Research Methods*:1–18, 2024.

[42] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. LINE: large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015*, pages 1067–1077. ACM, 2015.

[43] S. Thakoor, C. Tallec, M. G. Azar, R. Munos, P. Veličković, and M. Valko. Bootstrapped representation learning on graphs. In *ICLR 2021 Workshop on Geometrical and Topological Representation Learning*, 2021.

[44] P. Tovstogan, D. Bogdanov, and A. Porter. Mediaeval 2021: emotion and theme recognition in music using jamendo. In *Working Notes Proceedings of the MediaEval 2021 Workshop*, volume 3181 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2021.

[45] W. Trost, T. Ethofer, M. Zentner, and P. Vuilleumier. Mapping aesthetic musical emotions in the brain. *Cerebral cortex*, 22(12):2769–2783, 2012.

[46] S. Vashishth, S. Sanyal, V. Nitin, and P. P. Talukdar. Composition-based multi-relational graph convolutional networks. In *8th International Conference on Learning Representations, ICLR 2020*. OpenReview.net, 2020.

[47] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[48] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm. Deep graph infomax. In *7th International Conference on Learning Representations, ICLR 2019*. OpenReview.net, 2019.

[49] H. Xue, L. Xue, and F. Su. Multimodal music mood classification by fusion of audio and lyrics. In *MultiMedia Modeling: 21st International Conference, MMM 2015*, pages 26–37. Springer, 2015.

[50] Y.-H. Yang and H. H. Chen. *Music emotion recognition*. CRC Press, 2011.

[51] Y.-H. Yang, Y.-C. Lin, Y.-F. Su, and H. H. Chen. A regression approach to music emotion recognition. *IEEE Transactions on audio, speech, and language processing*, 16(2):448–457, 2008.

[52] J. Yang. A novel music emotion recognition model using neural network technology. *Frontiers in Psychology*, 12:760060, 2021.

[53] S. Zad, M. Heidari, H. James Jr, and O. Uzuner. Emotion detection of textual data: an interdisciplinary survey. In *2021 IEEE World AI IoT Congress (AIIoT)*, pages 0255–0261. IEEE, 2021.

[54] M. Zentner, D. Grandjean, and K. R. Scherer. Emotions evoked by the sound of music: characterization, classification, and measurement. *Emotion*, 8(4):494–521, 2008.

[55] K. Zhang, H. Zhang, S. Li, C. Yang, and L. Sun. The pmemo dataset for music emotion recognition. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval, ICMR 2018*, pages 135–142, 2018.

[56] L. Zhang, X. Yang, Y. Zhang, and J. Luo. Dual attention-based multi-scale feature fusion approach for dynamic music emotion recognition. In *Proceedings of the 24th International Society for Music Information Retrieval Conference, ISMIR 2023*, pages 207–214, 2023.

[57] Z. Zhou and M. Li. Semi-supervised regression with co-training. In *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 908–916. Professional Book Center, 2005.

[58] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation, 2002.