MOTIF-BASED GRAPH REPRESENTATION LEARNING FOR RECOMMENDER SYSTEMS

A THESIS SUBMITTED TO AUCKLAND UNIVERSITY OF TECHNOLOGY
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Supervisor

Jian Yu

Ji Ruan

Minh Nguyen

26 November 2025

By

Yuqi Zhang

School of Engineering, Computer and Mathematical Sciences

Abstract

While graph-based collaborative filtering makes use of the rich user–item relational structure in recommender systems, standard graph neural networks focus mainly on pairwise dependencies, which limits their ability to capture higher-order information that strongly affects recommendation accuracies. This thesis studies how to make graph-based collaborative filtering more robust and powerful by injecting network motifs into graph neural networks. We formalise common bipartite motifs with up to four nodes and demonstrate their integration into representation learning. Building on this, we propose four methods. First, MGSR introduces a motif-level attention mechanism that aggregates motif-induced neighbours in a single layer, which captures higher-order context and outperforming strong baselines on ProgrammableWeb dataset. Second, to scale motif usage beyond small graphs, we design fast generation algorithms for motif adjacency matrices and a lightweight MotifGCN that integrates these matrices into propagation with no extra parameters. Our experiments show that MotifGCN outperform state-of-the-art baselines on four real-world datasets. Through theoretical analysis, we demonstrate that motifs alleviate over-squashing compared with standard layer-wise propagation on original adjacency matrix. Third, we develop MGGCL, a motif-guided contrastive framework that constructs two motif-augmented views so that contrast learning emphasises stable co-interaction signals and alleviates popularity bias introduced by high-degree nodes. Experiment results show consistent performance improvement over baselines, especially on skewed datasets. Finally, we address

heterophily with MoHeGCL, which treats each user-anchored triad as a supervision unit, assigns soft homophily labels online, and switches between feature alignment and separation per neighbourhood through heterophily-aware propagation. This improves ranking quality while keeping training costs comparable to SimGCL. Together, these components demonstrate that motif-aware modelling offers a principled way to strengthen message passing, design interpretable self-supervision, and adapt to graphs with different homophily level. We also discuss limitations of our research and outline several future directions.

Contents

List of Figures

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning.

Signature of candidate

Publications

Zhang, Y., Wang, N., Yu, J., Yongchareon, S., & Nguyen, M. (2022). A Short Survey on Inductive Biased Graph Neural Networks. 2022 International Conference on Service Science (ICSS), 64–71. https://doi.org/10.1109/ICSS55994.2022.00019

Zhang, Y., Yu, J., Ruan, J., Wang, N., Madanian, S., & Wang, G. (2023). Motif-based Graph Attention Network for Web Service Recommendation. Proceedings of the 2023 Australasian Computer Science Week, 143–146. https://doi.org/10.1145/3579375.3579393

Wang, G., Yu, J., Nguyen, M., Zhang, Y., Yongchareon, S., & Han, Y. (2023). Motif-based graph attentional neural network for web service recommendation. Knowledge-Based Systems, 269, 110512. https://doi.org/10.1016/j.knosys.2023.110512

Zhang, Y., Yu, J., Liu, Z., Wang, G., Nguyen, M., Sheng, Q. Z., & Wang, N. (2025). Improving graph collaborative filtering with network motifs. Neural Computing and Applications, 1–20. https://doi.org/10.1007/s00521-025-11079-8

Zhang, Y., Yu, J., Wang, G., Liu, Z., Nguyen, M., Yang, L., & Wang, N., (2025). MGGCL: Motif-Guided Graph Contrastive Learning for Recommendation. Under Review.

Zhang, Y., Yu, J., & Wang, N., (2025). Motif-Based Heterophily-Aware Graph Contrastive Learning for Recommendation. Under Review.

Acknowledgements

I would like to express my deepest gratitude to my primary supervisor, Associate Professor Jian Yu, whose outstanding guidance, encouragement, and constant support have been central to the completion of this PhD. His insight, patience, and dedication have shaped both the direction of this research and my development as a scholar.
I am also sincerely grateful to my secondary supervisors, Dr. Ji Ruan and Professor Minh Nguyen, for their invaluable contributions throughout this journey. Their expertise, thoughtful feedback, and steady support greatly enriched this thesis, and I deeply appreciate the time and care they invested in my work.
My thanks further extend to my colleagues and friends at AUT. Finally, I am grateful to my family for their unconditional love and support throughout my PhD journey.

Chapter 1

Introduction

1.1 Introduction

Recommender systems have become essential in modern society, where people are easily overwhelmed by vast and often noisy online information originating from news platforms, social networks, entertainment services, e-commerce sites, and a wide range of other digital services. By modelling user preferences and predicting relevance, these systems make customised recommendations that users are more likely interested so that user experience is enhanced while platform can also improve their operational efficiency. Traditional recommender systems primarily rely on two paradigms: content-based filtering and collaborative filtering.

Content-based methods (M. Pazzani & Billsus, 1997; Billsus & Pazzani, 2000) represent items using their descriptive attributes, e.g. genres, tags, or textual descriptions, and recommend items that are similar to those a user has liked before, thus personalising suggestions based on individual preference profiles. Collaborative filtering instead exploits patterns of co-interaction across the entire user community. User-based approaches (Resnick, Iacovou, Suchak, Bergstrom & Riedl, 1994) identify users that share similar preferences and recommend items that are interesting to them, while item-based

approaches (Sarwar, Karypis, Konstan & Riedl, 2001) recommend items that tend to be consumed together. Later, matrix factorisation models (Mnih & Salakhutdinov, 2008; Koren, Bell & Volinsky, 2009; Rendle, 2010; X. He, Zhang, Kan & Chua, 2017) extended collaborative filtering by learning low-dimensional latent vectors for users and items, allowing the system to generalise beyond direct co-occurrence and handle large sparse rating matrices more effectively. Despite their success and wide deployment, these traditional methods largely focus on pairwise interactions in a user–item matrix and struggle to fully capture complex higher-order relational structures, contextual factors, and heterogeneous side information that are increasingly common in recommendation scenarios.

To address these limitations, there has been growing attention toward graph-based representation learning, where user–item interactions are naturally modelled as bipartite graphs. Within this paradigm, Graph Neural Networks (GNNs) have emerged as a powerful tool, leveraging message passing to aggregate information from direct and indirect neighbourhoods and encode complex collaborative signals. As a type of deep learning models tailored for graph-structured data, GNNs iteratively refine node representations by aggregating, transforming, and propagating information along edges and enables the flexible modelling of complex non-Euclidean relationships (T. N. Kipf & Welling, 2017; Z. Wu et al., 2021). Graph-based recommender systems, such as NGCF (X. Wang, He, Wang, Feng & Chua, 2019) and subsequent variants (X. He et al., 2020; J. Wu et al., 2021), have demonstrated significant performance improvements by exploiting structural signals that go beyond what traditional models can express.

However, several practical gaps remain to be explored. First, supervision in recommendation is sparse and noisy, which makes models sensitive to how we propagate signals across the user–item graph and to how we regularise embeddings. Over-smoothing across many layers can blur user and item distinctions, especially around high-degree hubs (K. Xu et al., 2018). These issues are amplified when homophily level varies.

Some user-item connections follow the homophilic assumption where nodes with similar features tend to connect, while others span diverse categories so standard graph convolution is not always appropriate. In addition, existing GNN recommenders typically rely on simple adjacency structures and uniform neighbourhood aggregation, which often overlook meaningful higher-order structures embedded in user–item graphs. In many real-world systems, interactions exhibit higher-level organisational patterns that cannot be fully expressed through pairwise edges alone.

Defined as small and recurring local patterns in graphs, network motifs provide a principled way to capture these higher-order dependencies (Milo et al., 2002). Motifs encode functional and structural semantics, revealing how groups of users or items co-occur, cluster, or interact in different characteristic ways. Incorporating motifs into GNN-based recommender systems offers a promising direction to enhance the expressiveness of graph representations, improve robustness in sparse conditions, and better reflect the underlying relational dynamics of user behaviour. Motif adjacency matrices can connect nodes that co-occur in informative patterns and thus help a single layer capture richer context that otherwise needs multiple-layer propagations (Benson et al., 2016). Nevertheless, leveraging motifs effectively within modern GNN architectures remains an open challenge, which is a motivation for further development of new models that integrate higher-order structural priors with graph learning mechanisms.

Self-supervised graph contrastive learning (GCL) has been adopted to improve robustness under sparsity (J. Wu et al., 2021; K. Zhou et al., 2020; Z. Lin, Tian, Hou & Zhao, 2022; J. Yu et al., 2022). However, many GCL pipelines still rely on simple random augmentations that can delete meaningful interactions and distort recommendation semantics, limiting interpretability and sometimes degrading performance (H. Liang et al., 2023; Trivedi, Lubana, Yan, Yang & Koutra, 2022; C. Wei et al., 2023). There is still no consensus on how to construct graph views that respect the user–item bipartite structure and differentiate strong and weak signals. Real recommendation graphs are

also highly skewed. Popular items and users create dense star-shaped patterns that can dominate learning, while structurally richer co-interaction structures are under-represented. Methods that do not explicitly counteract this imbalance risk reinforcing popularity bias and overlooking fine-grained preferences.

A further open issue is heterophily. In practice, users often bridge disparate item categories, and items attract heterogeneous audiences. Classical graph learning typically assumes homophily, where connected nodes tend to share similar labels or features, but many real-world graphs exhibit substantial heterophily, where neighbours are dissimilar and edges encode complementary or even opposite roles (J. Zhu et al., 2020; Pei, Wei, Chang, Lei & Yang, 2019; Chien, Peng, Li & Milenkovic, 2021). Standard smoothing helps in homophilic areas but can harm in heterophilic ones. Conversely, anti-smoothing may separate signals that should be pooled. Although heterophily-aware propagation and filtering have been explored in general GNNs, their systematic use in recommender systems remains limited.

Taken together, these observations bring the following motivations for our research on graph CF:

1. Explicit use of network motifs to strengthen signal under sparsity while remaining scalable.

2. Contrastive augmentations that are semantically faithful to recommendation graphs rather than uniformly random.

3. Propagation and objectives that adapt to local homophily or heterophily.

Addressing these needs would make representation learning more robust and interpretable for real-world recommendation scenarios.

1.2 Background

This section provides the necessary background for understanding the evolution of recommender systems, from traditional approaches to recent graph-based and contrastive learning methods, and outlines the key concepts that underpin the proposed framework in this thesis.

1.2.1 Recommender Systems

Recommender systems have become an essential component of modern digital platforms, enabling personalised access to large-scale collections of information, products, and services (Adomavicius & Tuzhilin, 2005; Ricci, Rokach & Shapira, 2011). They address the increasing challenge of information overload by predicting which movies, books, APIs, services, or news are most relevant to a specific user. The importance of recommender systems spans across domains including e-commerce, entertainment, social media, education, and web services. By learning patterns from historical interactions or contextual signals, these systems help improve user satisfaction, increase engagement, and optimise business outcomes such as click-through rates, conversions, and retention. As a result, recommender systems have evolved from simple heuristics into machine learning frameworks capable of modelling complex user–item relationships (S. Zhang, Yao, Sun & Tay, 2019).

Core Tasks and Data Characteristics

Most recommender systems target one or more of the following tasks (Ricci et al., 2011; S. Zhang et al., 2019):

- **Rating Prediction**. The goal is to predict a numerical score, e.g., a 1–5 star rating. This task dominated early recommender research but is now less emphasised in industrial practice.

- **Top-N Item Ranking**. The system recommends a ranked list of items predicted to be most relevant for each user. This is the most widely adopted task in modern systems.
- **Link Prediction in User–Item Graphs**. Under graph representation, interactions are treated as links. The goal becomes identifying the missing edges that represent potential relevance.
- **Sequential and Session-Based Recommendation**. These tasks model user behaviour over time. Sequential approaches consider long-term histories, while session-based methods focus on short-term patterns in anonymous or temporary sessions.
- **Context-Aware Recommendation**. Here, contextual attributes such as location, time, device, or domain-specific signals are integrated for more personalised prediction.
- **Cold-Start and Few-Shot Recommendation**. These tasks handle new users or items with limited historical data. Solutions often rely on content features, meta-learning, transfer learning, or self-supervised signals.

In these tasks, recommendation data typically exhibit the following characteristics (Y. Hu, Koren & Volinsky, 2008; X. He, Liao et al., 2017).

- **Sparse**. Most users interact with only a small subset of items.
- **Predominance of implicit feedback over explicit ratings.** Implicit feedback in the forms of clicks, views, or purchases are substantially more common than numeric ratings.
- **Long-tailed**. A small number of popular items generate most interactions.

- **Noisy**. User behaviour is influenced by randomness, biases, and external factors.
- **Heterogeneous**. Items may have associated text, images, knowledge graph relations, or contextual metadata.
- **Structured as graphs**. The user–item bipartite graph often captures higher-order patterns like motifs, meta-paths and communities that are not visible in flat matrices.

These characteristics motivate the development of models that can capture non-Euclidean structure, model uncertainty, and exploit rich auxiliary information.

Evaluation Protocols and Metrics

Evaluating recommender systems requires rigorous protocols to ensure fairness, reproducibility, and relevance to real-world environments (Herlocker, Konstan, Terveen & Riedl, 2004). Traditional random splits may not reflect temporal or sequential dependencies, which makes it essential to adopt more realistic strategies(Quadrana, Cremonesi & Jannach, 2018).

Common evaluation protocols include:

- **Random Hold-Out Splitting**(Herlocker et al., 2004; L. Wu, He, Wang, Zhang & Wang, 2023). User–item interactions are randomly divided into training, validation, and test sets. While simple and widely used, this approach ignores temporal order and may not fully reflect real-world recommendation behaviour.
- **Leave-One-Out (LOO) Evaluation** (X. He, Liao et al., 2017; Z. Sun et al., 2020). For each user, the last interaction is used for testing and the second-last for validation. This protocol is widely used for top-N recommendation due to its efficiency and consistency with consumer behaviour.

- **Temporal Splitting** (Hidasi, Karatzoglou, Baltrunas & Tikk, 2016; Quadrana et al., 2018). Interactions are ordered chronologically, ensuring that training data precede test data. This is crucial for sequential and real-world systems.

- **Session-Aware Evaluation** (Hidasi et al., 2016; Ludewig & Jannach, 2018). Individual sessions are split into train/test segments to evaluate session-based recommenders.

Based on the above evaluation protocols, performance is assessed using ranking metrics that reflect both relevance and ordering quality. Let U denote the set of users, I the set of items, $R_u \subseteq I$ the set of ground-truth relevant items for user u, and $\hat{R}_u@K$ the top-K items recommended to user u. Unless otherwise stated, all metrics are averaged over users.

- **Hit Ratio (HR@K)**. Under the LOO setting, each user $u \in U$ has exactly one ground-truth relevant item, denoted by $R_u = \{i_u\}$. Hit Ratio@K evaluates whether this item appears in the top-K recommendations $\hat{R}_u@K$.

$$\mathrm{HR}@K = \frac{1}{|U|} \sum_{u \in U} \mathrm{hit}(u).$$

where $\mathrm{hit}(u)$ is used determine whether a hit occurs:

$$\mathrm{hit}(u) = \begin{cases} 1, & \text{if } R_u \cap \hat{R}_u@K \neq \varnothing, \\ 0, & \text{otherwise}, \end{cases}$$

- **Recall@K**. Captures the proportion of relevant items retrieved in the top-K:

$$\mathrm{Recall}@K = \frac{1}{|U|} \sum_{u \in U} \frac{|R_u \cap \hat{R}_u@K|}{|R_u|}.$$

where $|R_u \cap \hat{R}_u@K|$ is the number of relevant items retrieved for user u, and $|R_u|$ is the total number of relevant items for user u. In LOO settings, Recall@K is equivalent to HR@K.

- **Precision@K**. Measures the fraction of recommended items in the top-K that are relevant:

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

Here, K is the recommendation list length, and $|R_u \cap \hat{R}_u@K|$ is the number of relevant items within the top-K for user u.

- **Normalized Discounted Cumulative Gain (NDCG@K)**. Evaluates ranking quality by rewarding relevant items at higher positions and discounting lower ranks. For user u, the discounted cumulative gain (DCG) is:

$$\text{DCG@}K_u = \sum_{i \in \hat{R}_u@K} \frac{2^{y_{u,i}} - 1}{\log_2\big(\text{rank}(u,i) + 1\big)},$$

where $y_{u,i} \in \{0, 1\}$ indicates whether item i is relevant to user u, and $\text{rank}(u, i)$ is the position of item i in u's ranked list (with 1 being the top rank). The normalised NDCG@K is:

$$\text{NDCG@}K = \frac{1}{|U|} \sum_{u \in U} \frac{\text{DCG@}K_u}{\text{IDCG@}K_u},$$

where $\text{IDCG@}K_u$ is the DCG of an ideal ranking in which all relevant items for user u are placed at the top positions.

- **Mean Average Precision (MAP)**. Average Precision (AP) for user u aggregates precision at ranks where relevant items appear:

$$\text{AP}_u = \frac{1}{|R_u|} \sum_{k=1}^{K} \text{Precision@}k_u \cdot y_{u,k},$$

where $\mathrm{Precision@}k_u$ is the precision at cut-off k for user u, and $y_{u,k}$ indicates whether the item at position k is relevant ($y_{u,k} = 1$) or not ($y_{u,k} = 0$). The Mean Average Precision is:

$$\mathrm{MAP} = \frac{1}{|U|} \sum_{u \in U} \mathrm{AP}_u.$$

where AP_u is the average precision score for user u, and the outer average aggregates over all users.

- **Coverage**. Assesses how widely the algorithm utilises the item catalogue:

$$\mathrm{Coverage} = \frac{\left| \bigcup_{u \in U} \hat{R}_u@K \right|}{|I|}.$$

where $\bigcup_{u \in U} \hat{R}_u@K$ is the set of distinct items that appear in at least one user's top-K list, and $|I|$ is the total number of items.

- **Novelty.** Encourages recommending fewer popular items. A common formulation is based on self-information:

$$\mathrm{Novelty@}K = \frac{1}{|U|} \sum_{u \in U} \frac{1}{K} \sum_{i \in \hat{R}_u@K} -\log_2 p(i),$$

where $p(i)$ is the empirical popularity of item i (e.g., the probability that i is interacted with across all users).

- **Diversity**. Evaluates how dissimilar the items in a recommendation list are from each other:

$$\mathrm{Diversity@}K = \frac{1}{|U|} \sum_{u \in U} \frac{2}{K(K-1)} \sum_{\substack{i,j \in \hat{R}_u@K \\ i<j}} \big(1 - \mathrm{sim}(i,j)\big),$$

where $\mathrm{sim}(i,j)$ denotes a similarity measure between items i and j (e.g., cosine similarity in an embedding space).

To ensure robustness, researchers typically report multiple metrics across various cut-off values K and compare against strong baselines. In addition, ablation studies are conducted to disentangle the contributions of individual model components.

Traditional and Modern Approaches

Early recommender systems were dominated by content-based filtering (CBF) and collaborative filtering (CF) (Adomavicius & Tuzhilin, 2005; Ricci et al., 2011). Content-based methods recommend items that are similar to those that a user has previously liked, assuming that past interests are useful predictors of future preferences (M. J. Pazzani & Billsus, 2007; Lops, de Gemmis & Semeraro, 2011). In these models, items are typically represented as feature vectors constructed from their attributes, such as descriptions, genres, keywords, tags, or other metadata. For users, their profiles are learned by aggregating or weighting the features of items the user has consumed. Recommendation is then made by measuring cosine similarity or other distance metrics between the candidate items and the user profile (Adomavicius & Tuzhilin, 2005; Ricci et al., 2011). This type of approaches have several advantages: it is relatively interpretable, can naturally handle new items as long as their content features are available, and does not require information from other users (Lops et al., 2011).

CBF also has several notable limitations. First, it tends to recommend items that closely resemble a user's historical choices, which leads to over-specialisation and reduced novelty (M. J. Pazzani & Billsus, 2007; Ricci et al., 2011). Second, CBF is strongly dependent on the availability and quality of item side features and on sufficiently rich user histories. When item descriptions are sparse, noisy, or fail to capture abstract or nuanced aspects of utility, the model struggles to represent complex preferences (Lops et al., 2011). Similarly, it becomes difficult to construct reliable profiles for users with very limited interaction history, which aggravates the cold-start problem (Adomavicius & Tuzhilin, 2005). Third, because CBF typically relies on

item–item or item–profile similarities, its performance is highly sensitive to the choice and tuning of similarity metrics. Last but not least, CBF does not naturally exploit cross-user behavioural patterns, and thus may miss collaborative signals that emerge only at the population level (Adomavicius & Tuzhilin, 2005; Ricci et al., 2011).

CF methods have become highly influential because it leverages "community wisdom" rather than relying solely on item attributes (Adomavicius & Tuzhilin, 2005; Ricci et al., 2011). CF methods can be divided into three categories: memory-based, model-based, and hybrid-based methods (Su & Khoshgoftaar, 2009; Ricci et al., 2011). Memory-based methods can be further split into user-based CF, item-based CF and neighbourhood-based CF. In user-based CF, the central idea is that users who have similar rating or interaction patterns in the past are likely to share similar preferences in the future (Resnick et al., 1994; Breese, Heckerman & Kadie, 1998). The system first computes pairwise user–user similarities, and then predicts a target user's preference for an item by aggregating the ratings or implicit feedback of their most similar neighbours. Item-based CF focuses on similarities between items, assuming that if a user liked an item, they may also like other items that tend to be co-consumed by many users (Sarwar et al., 2001). Here, item–item similarity matrices are precomputed, and recommendations are generated by combining the similarities between items in the user's history and candidate items. Neighbourhood-based CF methods are intuitive and relatively easy to implement. They often perform well in moderately sized dense datasets (Su & Khoshgoftaar, 2009). Moreover, they can provide some degree of explainability, for example through statements like "users similar to you also liked…" (Herlocker, Konstan & Riedl, 2000).

However, these approaches also have significant drawbacks. They are highly sensitive to data sparsity, since reliable similarity estimates require sufficient overlap in user–item interactions (Sarwar et al., 2001; Adomavicius & Tuzhilin, 2005). When the user–item matrix contains many missing values, similarity computations become

noisy or unreliable which causes significant performance degradation. In addition, neighbourhood-based CF can suffer from scalability issues, as computing and maintaining large similarity matrices becomes expensive with growing numbers of users and items (Sarwar et al., 2001; Su & Khoshgoftaar, 2009). These methods also tend to overemphasise popular items and reinforce popularity bias. They provide limited support for cold-start users or items as they rely entirely on historical interaction patterns without incorporating auxiliary content or contextual information (Ricci et al., 2011; Lü et al., 2012).

A major breakthrough came with matrix factorisation (MF) models, which rose to prominence during the Netflix Prize competition (Bennett & Lanning, 2007; Koren et al., 2009). The core idea of MF is to represent both users and items in a shared low-dimensional latent space, where each user and item is associated with a learned embedding vector. Instead of relying on explicit similarity computations, MF decomposes the sparse user–item interaction matrix into two dense matrices: one capturing user latent factors and the other capturing item latent factors (Koren et al., 2009). A user's preference for an item is then estimated by taking the inner product of their corresponding latent embeddings. This inner product reflects how well the user's latent preferences align with the item's latent characteristics. During training, the model adjusts these embeddings so that observed interactions receive high predicted scores, while unobserved or negative interactions receive lower scores, typically using optimisation techniques such as stochastic gradient descent with L2 regularisation (Koren, 2008; Koren et al., 2009). This framework offers several advantages. By projecting high-dimensional and sparse data into a compact latent space, MF captures hidden structures such as latent user preference factors and item attributes that are not explicitly available in the metadata (Koren et al., 2009). It also generalises more effectively than neighbourhood-based methods because it does not depend on overlapping interactions between pairs of users or items. Instead, it learns a global set of factors that explain

patterns across the entire dataset.

MF has inspired numerous influential extensions. Probabilistic Matrix Factorisation introduces a probabilistic framework with Gaussian priors to improve robustness under sparsity (Salakhutdinov & Mnih, 2007). SVD++ incorporates implicit feedback, such as clicks or views, to better model user engagement (Koren, 2008). Factorisation Machines further generalise MF by enabling pairwise interactions between arbitrary features, making it effective when side information is available (Rendle, 2010). Owing to its simplicity, efficiency and strong performance, MF remains one of the most enduring and widely used classical baselines for personalised recommendation (Ricci et al., 2011).

In the last decade, recommender systems have undergone a major transformation, shifting from linear models and similarity-based methods towards deep learning and graph-based approaches (S. Zhang et al., 2019). Deep neural networks offer the ability to model complex, nonlinear relationships between users and items that traditional matrix factorisation or neighbourhood-based methods cannot easily capture (Covington, Adams & Sargin, 2016; X. He, Liao et al., 2017). Early deep learning recommenders employed multilayer perceptrons (MLPs) to learn nonlinear user–item interaction functions directly from embeddings, enabling richer representations of collaborative signals (X. He, Liao et al., 2017). Subsequent work expanded to convolutional neural networks (CNNs), which are effective for sequence-aware recommendation by capturing local patterns and short-term behavioural dynamics in user interaction histories (J. Tang & Wang, 2018; F. Yuan, Karatzoglou, Arapakis, Jose & He, 2019).

Recurrent neural networks (RNNs) and their variants, such as LSTMs and GRUs, extended this paradigm by modelling temporal dependencies and long-range sequential patterns, making them well suited for session-based or time-sensitive recommendation tasks (Hidasi et al., 2016; Hidasi & Karatzoglou, 2018). Later, the incorporation of attention mechanisms has further advanced the field by allowing models to focus selectively on the most informative interactions or past behaviours (G. Zhou et al.,

2018; C. Wu et al., 2019). Attention-based models, particularly those inspired by the Transformer architecture, can capture both short-term and long-term dependencies without the constraints of fixed receptive fields or sequential processing (W.-C. Kang & McAuley, 2018; F. Sun et al., 2019). This enables more expressive modelling of user intent, contextual signals, and heterogeneous interaction patterns. Such developments have significantly broadened the modelling capacity of recommender systems, paving the way for more adaptive, personalised, and context-aware recommendation strategies (S. Zhang et al., 2019; Y. Zhang & Chen, 2020).

Graph-based recommender systems, particularly those built on Graph Neural Networks (GNNs), represent one of the most significant modern advances in recommendation research (S. Zhang et al., 2019; Z. Wu et al., 2021). Unlike traditional models that treat the user–item interaction matrix as an unstructured collection of entries, graph-based methods explicitly model interactions as a bipartite graph, where users and items are nodes connected by edges denoting observed behaviours (Berg, Kipf & Welling, 2017; Ying et al., 2018). This graph formulation allows GNNs to perform message passing which iteratively aggregates information from a node's neighbours to refine its representation (T. N. Kipf & Welling, 2017; Z. Wu et al., 2021). By propagating signals across multi-hop neighbourhoods, GNNs capture higher-order collaborative patterns that are difficult to infer from pairwise interactions alone (X. Wang, He, Wang et al., 2019; X. He et al., 2020). Representative GNN-based recommender systems such as NGCF, LightGCN, and numerous subsequent extensions exploit these structural signals to learn rich, topology-aware embeddings that substantially outperform matrix factorisation and deep learning baselines (X. Wang, He, Wang et al., 2019; X. He et al., 2020; W. Yu, Zhang & Qin, 2022). Detailed introduction to GNNs will be covered later in Section 1.2.2.

Building on the success of GNNs, recent research has introduced Graph Contrastive Learning (GCL) as a powerful paradigm for improving graph-based recommendation

under sparse, noisy, or incomplete data (Z. Wu et al., 2021; Y. Liu, Jin et al., 2023). GCL applies principles of self-supervised learning by generating perturbed "views" or augmentations of the interaction graph and training the model to produce consistent representations across these views (You et al., 2020; Y. Liu, Jin et al., 2023). This encourages the GNN to learn features that are robust to structural variations while still preserving the semantic relationships inherent in user–item interactions. Approaches such as SGL, SimGCL, and other contrastive variants demonstrate that contrastive objectives can significantly improve generalisation, reduce overfitting, and alleviate sparsity by leveraging abundant unlabelled interactions (J. Wu et al., 2021; J. Yu et al., 2022; J. Yu, Yin et al., 2023). This shift reflects a broader movement from purely predictive GNN models toward self-supervised, augmentation-based representation learning, enabling recommender systems to exploit more information from the underlying graph structure without requiring additional labels (S. Zhang et al., 2019; Y. Liu, Jin et al., 2023; J. Yu, Yin et al., 2023). Detailed introduction to GCL methods will be covered later in Section 1.2.2.

1.2.2 Graph Neural Networks

Graph-structured data are widely used to represent complex relationships among entities in numerous real-world systems (M. Newman, 2018; Barabási, 2013). Traditional machine learning techniques, which typically assume independent and identically distributed (i.i.d.) samples, often struggle to capture the rich relational information inherent in graph data (Bronstein, Bruna, LeCun, Szlam & Vandergheynst, 2017; W. L. Hamilton, 2020). Graph Neural Networks (GNNs) have emerged as a powerful class of deep learning models designed specifically to operate on graphs and learn expressive node, edge, or graph-level representations through iterative message-passing mechanisms (Gilmer, Schoenholz, Riley, Vinyals & Dahl, 2017; Battaglia et al., 2018).

By integrating neural learning principles with graph theory, GNNs can effectively model high-dimensional non-Euclidean data and have achieved state-of-the-art performance across many domains (T. N. Kipf & Welling, 2017; W. Hamilton, Ying & Leskovec, 2017; Z. Wu et al., 2021).

GNNs are deep learning architectures that generalise the convolution operation from Euclidean data to non-Euclidean graph domains (Bronstein et al., 2017; Z. Wu et al., 2021). The core idea is that each node updates its representation by aggregating and transforming information from its neighbours (T. N. Kipf & Welling, 2017; W. Hamilton et al., 2017), a process often referred to as message passing, neighbourhood aggregation, or graph convolution. This process is repeated for multiple layers and it enables nodes to capture multi-hop structural dependencies (Gilmer et al., 2017; Battaglia et al., 2018).

Formally, a typical GNN layer can be described as:

$$h_v^{(k)} = \mathrm{UPDATE}^{(k)}\Big(h_v^{(k-1)}, \mathrm{AGGREGATE}^{(k)}\big(\{h_u^{(k-1)} : u \in \mathcal{N}(v)\}\big)\Big),$$

where $h_v^{(k)}$ is the representation of node v at layer k, and $\mathcal{N}(v)$ denotes its neighbours. Through this iterative process, GNNs learn to embed nodes into a latent vector space that encodes both node features and relational structure.

The early generation of GNNs, such as graph convolutional networks (GCNs) (T. N. Kipf & Welling, 2017) and GraphSAGE (W. Hamilton et al., 2017), introduced the foundational paradigm of neighbourhood aggregation. Subsequent advances extend GNNs with attention mechanisms (Veličković et al., 2018), positional encodings (Dwivedi, Luu, Laurent, Bengio & Bresson, 2022), spectral filters (Bronstein et al., 2017), and contrastive learning objectives (You et al., 2020). These developments make GNNs adaptable to a wide range of graph types, including homogeneous, heterogeneous, dynamic and knowledge graphs (Z. Zhang, Cui & Zhu, 2022).

GNN Applications

Since many real-world systems can naturally be represented as graphs where nodes denote entities and edges represent relationships, GNNs have been broadly adopted across scientific, industrial, and social domains (Bronstein et al., 2017; Z. Wu et al., 2021). Social platforms, for example, rely on graphs to model friendships, followers, interactions, and content propagation. On these graphs, GNNs support tasks such as friend recommendation, community detection, fake-news detection, and influence modelling (W. Hamilton et al., 2017; Veličković et al., 2018). By capturing higher-order connectivity patterns, they allow the system to understand how users relate to one another beyond direct connections.

In the context of structured knowledge and language, GNNs are widely used on knowledge graphs for completion, relation extraction, question answering, and entity disambiguation (Schlichtkrull et al., 2017; Ying et al., 2018). Their ability to propagate semantic information across multi-relational edges makes them well suited for reasoning over symbolic knowledge bases, where variants such as relational GNNs and graph attention networks perform contextualised inference over entities and relations (Schlichtkrull et al., 2017; Veličković et al., 2018).

GNNs have also become central in computational chemistry and drug discovery, where molecules are naturally represented as graphs of atoms and chemical bonds. GNN-based molecular models, including message-passing neural networks, now achieve state-of-the-art performance in predicting molecular properties, protein-ligand interactions, and quantum mechanical attributes (Duvenaud et al., 2015; Gilmer et al., 2017). Their structural awareness allows them to encode 3D interactions that are essential for chemical and biochemical reasoning. Similar ideas extend to physical modelling, where physically informed GNNs approximate complex dynamical systems by embedding conservation laws and energy functions into graph-based message passing (Battaglia,

Pascanu, Lai, Rezende & Kavukcuoglu, 2016; T. Kipf, Fetaya, Wang, Welling & Zemel, 2018).

Spatio-temporal applications form another key area. Modelling urban traffic flow, mobility patterns, and transportation systems requires handling both spatial topologies and temporal dynamics. Spatio-temporal GNNs combine graph convolutions with recurrent or attention mechanisms to forecast traffic congestion, travel times, and demand distributions (B. Yu, Yin & Zhu, 2018; Y. Li, Yu, Shahabi & Liu, 2018). In finance and security analytics, transaction networks, customer-merchant graphs, and stock correlation networks provide structured signals for fraud detection, risk modelling, portfolio optimisation, and trend forecasting, where GNNs detect anomalies by aggregating behavioural cues from connected entities (X. Wang, He, Cao, Liu & Chua, 2019; J. Wang, Zhang, Xiao & Song, 2022).

In summary, these examples illustrate the versatility of GNNs in modelling interconnected data across domains where relational structure and dependency patterns are fundamental.

GNN Architectures

GNN architectures can be broadly categorised into three families: spectral methods, spatial methods, and advanced variants that build upon the message-passing paradigm (Bronstein et al., 2017; Z. Wu et al., 2021).

Spectral approaches define convolution in the frequency domain of the graph Laplacian. Early models such as spectral CNNs (Bruna, Zaremba, Szlam & LeCun, 2014) and Chebyshev networks (Defferrard, Bresson & Vandergheynst, 2016) introduced polynomial filters that approximate spectral convolutions. The widely used GCN simplifies these operations with a renormalisation scheme, which yields efficient propagation (T. N. Kipf & Welling, 2017):

$$H^{(k)} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(k-1)}W^{(k)},$$

where $\tilde{A}$ is the adjacency matrix with self-loops, $\tilde{D}$ is the corresponding degree matrix, $H^{(k-1)}$ is the node representation matrix from the previous layer, and $W^{(k)}$ is a learnable weight matrix. GCNs are effective for semi-supervised classification but rely on fixed graph structures and typically lack inductive capabilities (T. N. Kipf & Welling, 2017; W. L. Hamilton, 2020).

Spatial approaches define convolutions directly on node neighbourhoods and in doing so generalise the local receptive field idea from convolutional neural networks to graph domain (W. Hamilton et al., 2017; Veličković et al., 2018). A generic spatial GNN layer can be written in a message-passing form as

$$h_i^{(k)} = \sigma\left(W^{(k)}h_i^{(k-1)} + \sum_{j\in\mathcal{N}(i)} \alpha_{ij}^{(k)}\, U^{(k)}h_j^{(k-1)}\right),$$

where $h_i^{(k)}$ is the representation of node i at layer k, $\mathcal{N}(i)$ denotes its neighbourhood, $W^{(k)}$ and $U^{(k)}$ are learnable weight matrices, $\alpha_{ij}^{(k)}$ is a normalisation or attention coefficient, and $\sigma(\cdot)$ is a nonlinear activation function.

GraphSAGE introduces sampling-based aggregation functions to enable inductive learning on large graphs, including mean, LSTM, and pooling aggregator (W. Hamilton et al., 2017). Graph Attention Networks (GAT) add attention mechanisms to assign learnable weights to neighbours which improves the expressiveness of neighbourhood aggregation (Veličković et al., 2018). Graph Isomorphism Networks (GIN) maximise representational power using injective aggregation functions (K. Xu, Hu, Leskovec & Jegelka, 2019). More generally, message-passing neural networks (MPNNs) provide a unified framework for defining message computation and update functions on nodes and edges (Gilmer et al., 2017).

Recent recommendation-oriented GNNs emphasise efficiency and scalability. LightGCN removes transformation matrices and nonlinear activation functions and retains only neighbourhood propagation and layer-wise embedding averaging (X. He et al., 2020). PinSage combines random walks with GraphSAGE-style aggregation for web-scale recommendation (Ying et al., 2018). Heterogeneous GNNs extend message passing to multi-relational graphs using meta-paths or meta-graphs (C. Zhang, Song, Huang, Swami & Chawla, 2019). Motif-based GNNs leverage higher-order structural patterns to encode semantic dependencies beyond pairwise edges (X. Chen et al., 2022). Hybrid architectures further combine GNNs with probabilistic models, attention mechanisms, transformers, or contrastive objectives to enhance robustness and capture complex interaction patterns (You et al., 2020; Y. Liu, Jin et al., 2023).

These diverse architectures showcase the flexibility of the GNN paradigm and its adaptability to specialised graph domains including large-scale recommender systems (X. Wang, He, Wang et al., 2019; Z. Wu et al., 2021).

Graph Neural Networks for Recommender Systems

Recommender systems aim to predict user preferences and provide personalised suggestions for items such as movies, products, services, or news (Adomavicius & Tuzhilin, 2005; Ricci et al., 2011). Traditional methods, including collaborative filtering, matrix factorisation, and content-based filtering, have contributed significantly to modern recommendation. However, these methods often rely on shallow architectures, assume linear interactions, or fail to capture higher-order connectivity patterns present in user-item bipartite graphs (X. He, Liao et al., 2017; S. Zhang et al., 2019).

GNNs have become highly influential in recommendation by offering a principled framework to model complex user-item relationships (Z. Wu et al., 2021). In a recommender system, nodes represent users and items, while edges represent observed

interactions such as ratings, purchases, or clicks. GNNs leverage neighbourhood aggregation to capture multi-hop collaborative signals (X. Wang, He, Wang et al., 2019; X. He et al., 2020). For example, a user's embedding can be influenced not only by items they directly consumed but also by the behaviour of similar users or related items.

Key benefits of using GNNs for recommendation include:

- **Higher-order connectivity modelling:** Multi-layer propagation captures indirect relationships and alleviates sparsity (X. Wang, He, Wang et al., 2019).
- **Contextual and semantic representation:** By integrating edge attributes like timestamps and ratings, or content features including text and images, GNNs produce richer embeddings (Ying et al., 2018).
- **Flexibility to incorporate side information:** Heterogeneous GNNs extend naturally to user attributes, item attributes, multiple interaction types or contextual graph structures (C. Zhang et al., 2019).
- **Compatibility with contrastive learning:** Modern graph contrastive learning frameworks further enhance embedding robustness by enforcing consistency between perturbed or augmented graph views (You et al., 2020; J. Wu et al., 2021; J. Yu et al., 2022).

Representative GNN-based recommendation models include GCN-based methods such as PinSage (Ying et al., 2018), NGCF (X. Wang, He, Wang et al., 2019), and LightGCN (X. He et al., 2020); GraphSAGE-based models for inductive user/item representation learning (W. Hamilton et al., 2017); GAT-based models for personalised attention over interactions (Veličković et al., 2018); heterogeneous GNNs that exploit meta-path or motif structures (C. Zhang et al., 2019; X. Chen et al., 2022); and graph contrastive learning models that apply augmentation and consistency regularisation (J. Wu et al., 2021; J. Yu et al., 2022).

Graph Contrastive Learning

Graph Contrastive Learning (GCL) has emerged as a powerful self-supervised learning tool for learning robust graph representations without requiring extensive labelled data. Traditional supervised GNNs optimise node or graph classification objectives, which rely on limited and often expensive annotations. In contrast, GCL draws inspiration from contrastive learning in computer vision and representation learning, aiming to maximise agreement between different augmented views of the same graph while distinguishing them from representations of other nodes or subgraphs (Veličković et al., 2019; You et al., 2020).

The core idea of GCL is to construct perturbations of the input graph and train the GNN encoder so that representations of corresponding components across these perturbed views remain consistent (You et al., 2020; J. Qiu et al., 2020). The perturbations are typically chosen from node dropping, edge dropping, feature masking, random walks, or noise injection. Formally, two correlated graph views $(\mathcal{G}_1, \mathcal{G}_2)$ are generated through stochastic augmentation functions, and the contrastive objective encourages the encoded representations to be close in latent space:

$$\mathcal{L}_{\text{GCL}} = -\sum_{i \epsilon \mathcal{V}} \log \frac{\exp\left(\text{sim}(z_i^{(1)}, z_i^{(2)})/\tau\right)}{\sum_{j \epsilon \mathcal{V}} \exp\left(\text{sim}(z_i^{(1)}, z_j^{(2)})/\tau\right)},$$

where $z_i^{(1)}$ and $z_i^{(2)}$ are the representations of node i in the two views, $\text{sim}(\cdot, \cdot)$ denotes cosine similarity, and τ is a temperature parameter. This InfoNCE-style loss (Oord, Li & Vinyals, 2019) encourages alignment of positive pairs and separation from negative samples, leading to embeddings that capture invariant structural and semantic properties of the graph.

GCL has been further extended with techniques that exploit global summaries, hierarchical graph structure, and hard negative sampling. Deep Graph Infomax (Veličković

et al., 2019) maximises mutual information between node embeddings and a global graph summary vector, while GraphCL (You et al., 2020) introduces domain-specific graph augmentations for improved generalisation. GCC (J. Qiu et al., 2020) learns transferable structural representations via contrastive objectives on subgraphs drawn from large unlabelled datasets.

In recommender systems, contrastive learning has proven particularly effective in mitigating sparsity and improving robustness to noise. Methods such as SGL (J. Wu et al., 2021) generate stochastic augmentations of the user–item interaction graph, while SimGCL (J. Yu et al., 2022) injects uniform embedding noise to simplify augmentation design. These approaches significantly enhance the quality of user and item embeddings by enforcing consistency across perturbed interaction patterns, which in turn captures high-order collaborative signals more effectively.

Overall, GCL represents a major advancement in self-supervised learning on graphs and enables GNNs to exploit unlabelled structural information. Its success in both general graph tasks and recommendation has established contrastive learning as a core component of modern graph representation learning.

1.2.3 Network Motifs

Network motifs are small, recurring, and statistically significant subgraph patterns such that their appearance in a given network is more frequent than in random networks (Milo et al., 2002). Unlike global statistical descriptors such as degree distributions or clustering coefficients, motifs capture localised interaction patterns that reveal functional, relational and behavioural regularities within networks.

Originally developed in the context of biological networks, motifs provided a means to identify building blocks of complex systems. For example, in gene-regulation networks of Escherichia coli, Shen-orr et al. found three highly significant motifs and

linked them to temporal expression programs and environmental responses (Shen-Orr, Milo, Mangan & Alon, 2002). Recently, a historical overview demonstrate that the concept of motifs and approaches of detecting motifs have been studied in ecology for decades (Stone, Simberloff & Artzy-Randrup, 2019).

Motifs have been studied across many fields (Milo et al., 2002, 2004; U. Alon, 2007). In biological and biochemical systems they help explain regulatory logic, robustness, modularity, and information-processing pathways (Shen-Orr et al., 2002; U. Alon, 2007). In ecological food webs, motif distributions reflect trophic relationships and energy flows (Stouffer, Camacho, Jiang & Nunes Amaral, 2007; Bascompte, 2009). In social networks and information systems, motifs such as open or closed triads characterise user interaction styles, community formation and trust propagation (Wasserman & Faust, 1994; M. Newman, 2018). For instance, a clustering triad motif in a social graph may indicate closure and homophily, whereas a "checker-board" motif in ecological networks may reflect competitive exclusion (Bastolla et al., 2009).

In engineered and technological networks, motifs capture essential structural design units. Power-grid networks and software dependency networks often reveal characteristic recurrences of certain sub-graphs that support reliability, redundancy or maintainability (Rosas-Casals, Valverde & Solé, 2007). Understanding motif distributions in these systems supports performance optimisation, fault diagnosis and resilient infrastructure design.

More recently, network motifs have found increasing relevance in machine learning and graph representation learning. Instead of only relying on node-level features or first-order adjacency, researchers have looked to embed motif-based features into systems such as graph neural networks or self-supervised graph frameworks. For example, in a research on molecular property prediction (Z. Zhang, Liu, Wang, Lu & Lee, 2021), the authors propose a motif-generation pre-training task to capture sub-graph semantics such as functional groups in molecular graphs. Similarly, motif-based methods have

been used to enhance GNN interpretability and expressivity (Z. Yu & Gao, 2022). Motif-aware attention networks also show that introducing motif-based higher-order structure can improve node-classification performance, as demonstrated in (Sheng, Zhang, Wang & Chang, 2024).

Compared with simple adjacency-based or neighbourhood-based representations, motifs provide richer semantic priors such as co-usage patterns, complementarity or competition relationships, and even heterophilic interactions where connected nodes are more likely to be dissimilar. In the context of recommender systems and service recommendation, motif-based structures allow one to capture multi-hop, multiplex or heterophilic patterns. An example is a biclique motif connecting users and items via multiple shared services. Such motifs are precisely aligned with higher-order dependencies, which cannot be well handled by simple GCN (T. N. Kipf & Welling, 2017) or LightGCN (X. He et al., 2020) architectures.

Recommendation scenarios are most commonly modelled as user–item bipartite graphs, where edges represent historical interactions such as ratings, clicks, or purchases. Within such bipartite graphs, motifs capture recurrent interaction patterns that go beyond simple user–item pairs, e.g., small subgraphs involving multiple users and items that co-occur in characteristic configurations. These higher-order structures reflect meaningful behavioural or semantic implications, such as groups of users who repeatedly consume similar item subsets or items that tend to be co-interacted with by overlapping user communities. Therefore, motifs provide richer signals than individual edges alone.

Motif information can be injected into recommendation models at multiple levels. Simple counts of motif instances can be used for node feature initialisation. Motif-induced adjacency matrices can define higher-order neighbourhoods for graph convolution, which allows message passing to reach to motif neighbours that are multi-hop away. In dynamic settings, temporal motif analysis can reveal emerging co-consumption patterns and enable models to adapt to evolving user interests and item relations.

As recommendation environments continue to grow in scale and complexity, network motifs offer a coherent and empirically grounded framework for modelling higher-order structure in recommendation graphs. Leveraging motif information is promising to improve accuracy, robustness, and interpretability across a wide range of graph-based recommender systems.

1.3 Research Questions

To bridge the motivations above with concrete inquiries, we focus on motifs as the thread that ties the gaps together. Different motifs capture complementary local structures in user–item graphs, but there is no existing method for representing and combining them within graph collaborative filtering. Even when motifs are useful, naive implementations can be too heavy for large-scale systems. Contrastive learning often relies on random perturbations that may break recommendation semantics, while motif structure offers a way to define more meaningful views. In addition, propagation in graph CF typically assumes homophily, whereas real-world data contain many heterophilic connections. Motif patterns can help decide when to smooth signals and when to keep them separate. These considerations lead to the following research questions.

1. How to combine various types of motifs into graph-based recommendation?
    - What types of motifs can be defined on recommendation graphs?
    - How to embed motifs' structural information into GNNs?
    - How to combine useful information in motifs while minimising the redundancy among similar motifs?
2. How can structural information in motifs be incorporated into graph CF with high efficiency while improving recommendation quality?

- How to design an efficient algorithm for generating motif adjacency matrices that can scale to large datasets?
- How do motifs impact on graph convolution in terms of propagation efficiency?
- How to design a motif-based graph neural network that have improved performance while remaining computationally efficient?

3. How can motifs guide the construction of contrastive views so that perturbations are semantically meaningful?
   - What types of motifs can be used for construction of contrastive views?
   - What are the semantic insights behind the contrast between motif-based contrastive views?
   - How can motif-based contrastive learning benefit recommendation?
4. How can motif-aware propagation or filtering make graph CF sensitive to heterophily in order to improve learning in regions where users and items connect across diverse categories?
   - How can motif-based structures be used to model and capture heterophilic relationships in recommendation graphs?
   - How can a graph CF model be designed to adapt to different homophily levels among every node's neighbourhood?

1.4 Contributions

This thesis has four contributions and each corresponds to a research question. First, we formalise how different motif types can be represented and fused within graph

collaborative filtering. Second, we introduce efficient approaches to deploy motifs at scale. Third, we use motif structures to design semantically meaningful contrastive views that avoid destructive random perturbations. Last, we leverage motif-aware propagation and filtering to make graph CF responsive to heterophily in real-world data. The following items summarise these contributions.

1. Addressing RQ1: We first explore the common motifs in the user-item bipartite graph and identify all seven motifs with up to four nodes in bipartite graphs. We then propose a motif attention mechanism that combines results on all motif adjacency matrices where on each motif adjacency matrix we run a single graph attention layer. Based on this mechanism, we then propose motif-based graph attention networks which dynamically learn weights of the motifs during training so the redundancy between different motif adjacency matrices is minimised.

2. Addressing RQ2: We first propose dedicated algorithms for generating motif adjacency matrices for all motifs consisting of up to four nodes in bipartite graphs. Theoretical analyses and experiments on two different sized datasets show that the algorithms have significantly improved time and space complexity over general approaches. Then, we propose a lightweight CF framework called MotifGCN, which integrates motifs in graph convolution layers to enhance the neighbourhood aggregation efficiency within each single layer without adding additional model parameters. Extensive experimental results on four real-world datasets show that MotifGCN significantly outperforms state-of-the-art baselines.

3. Addressing RQ3: We first formally establish a hierarchical relationship between the motif adjacency matrices constructed from star motifs and rectangle motifs. By explicitly contrasting the rectangle-motif view against the star-motif view, the model is optimised to strengthen node representations derived from these rectangle patterns where nodes have stronger correlation. We then present a novel

motif-guided graph contrastive learning framework that explicitly integrates structural semantics into graph view construction through motif-based augmentations. Extensive experiments on four real-world recommendation benchmarks demonstrate our model's consistent accuracy improvements over baselines, particularly on skewed datasets.

4. Addressing RQ4: We propose a motif-based heterophily-aware graph contrastive learning framework that treats each user-anchored triad as a contrast unit with soft homophily labels estimated online via a quantile-scheduled threshold. By estimating the proportions of different motif labels for each node, the contrastive objective is adaptive to the homophily level in the scope of local motif structure by reweighting propagation for each node. Through complexity analysis and experiments, we show that our proposed framework remains computationally efficient while improving recommendation performance on benchmark datasets.

1.5 Thesis Outline

The rest of this thesis is structured as follows.

Chapter 2 covers an in-depth literature review. It begins by introducing the foundational methods for recommender systems, i.e. content-based filtering and collaborative filtering. We then review traditional methods for mashup-API recommendation. In addition, we discuss existing research on network motifs and their applications in deep learning. Last but not least, we summarise existing research on GNNs in the following categories of ground-breaking GNNs, topology-aware GNNs, GNNs for collaborative filtering, GNNs for mashup-API recommendation, graph contrastive learning for collaborative filtering, motif-based GNNs and methods for collaborative filtering, and heterophily-aware GNNs and methods for collaborative filtering.

Chapter 3 - Chapter 6 are the main research components of this thesis. Each chapter addresses a research question by proposing corresponding algorithms or models with supporting experimental results.

Chapter 3 addresses RQ1 by studying how to represent and fuse multiple motif types in graph CF. We first review existing research on mashup-API service recommendation and motif-based graph neural networks. We then present a proposition for all seven motifs with up to four nodes in bipartite graphs and provide a sketch proof of the proposition. We then introduce the motif attention mechanism, which adapts the original graph attention mechanism to motif-aware scenario so the model can learn from each different motif structure. Based on this mechanism, we then propose motif-based graph attention networks and demonstrate its performance with experiment on the ProgrammableWeb.com dataset.

Chapter 4 addresses RQ2 with efficient motif adjacency matrix generation algorithms and lightweight motif-based graph CF model. We first review existing research on GNNs for CF, motif-based GNNs and higher-order enhanced GNNs for CF. We then introduce the background knowledge of network motifs and how motif adjacency matrices are generated. Next, we present our dedicated algorithms for generating motif adjacency matrices of motifs consisting of up to four nodes in bipartite graphs, with a complexity analysis against general approaches. We then propose the novel framework for CF called MotifGCN. Through experiments results for the algorithms and MotifGCN, we demonstrate superior performance of our proposed methods.

Chapter 5 addresses RQ3 by constructing motif-guided graph contrastive learning for CF. We first review existing research on GCL recommendation models that use simple random augmentations for contrastive views, methods that use structured and semantically guided augmentations, and motif-based approaches for graph recommendation and contrastive learning. We then discuss the background knowledge of motifs in recommendation graphs and contrastive learning for recommender systems.

After that, we thoroughly explain our motivation with a formal proposition of the structural hierarchy between motif adjacency matrices used in our method and provide a proof for the proposition. Based on this, we then present the architecture of our motif-guided graph contrastive learning framework, followed by performance comparison with state-of-the-art baselines on four public benchmark datasets and parameter sensitivity analysis.

Chapter 6 addresses RQ4 with a motif-based heterophily-aware graph contrastive learning framework for CF. We first review existing research on GCL for CF, motif-based GNNs for recommender systems and contrastive learning, and heterophily-aware GNNs for recommender systems. We then present the architecture of our proposed motif-based heterophily-aware graph contrastive learning framework with detailed explanation of each component, and provide an algorithm showing the whole training process. We then show the experiment results of performance comparison with baseline methods as well as the ablation studies.

Chapter 7 concludes the thesis by summarising the main contributions, acknowledging limitations, and outlining directions for future research.

Chapter 2

Literature Review

2.1 Introduction

This chapter reviews the body of literature that forms the foundation of this thesis. It begins with foundational recommender-system methods including content-based approaches and collaborative filtering techniques, as well as their deep learning extensions. This establishes the fundamental principles of user-item modelling and highlights limitations such as sparsity and cold-start issues. We then examine research on mashup-API recommendation, a domain relevant to service composition in Web-based software ecosystems. Next, we introduce network motifs as higher-order patterns that capture recurring substructures in graphs. Finally, we review graph neural networks (GNNs), including fundamental architectures, topology-aware variants, and GNN-based methods for collaborative filtering. Recent advancements in graph contrastive learning and motif-based or heterophily-aware GNNs illustrate the growing interest in complexer graph representations, and reveal research gaps that this thesis aims to address.

2.2 Foundations of Recommender System

Recommender systems are now an essential part of modern information platforms (Ricci et al., 2011). They give users personalised access to data spaces that are increasingly large and complex. As users interact with digital services such as electronic commerce platforms, media streaming sites, social networks and API marketplaces, they generate behavioural signals that can be used to infer preferences and deliver relevant suggestions. The main goal of a recommender system is to model these preferences and predict the items, services or content that users are most likely to engage with. Over the past two decades this field has evolved from heuristic based approaches to sophisticated machine learning pipelines that can capture high dimensional patterns, contextual factors and dynamic user behaviour (Adomavicius & Tuzhilin, 2005). Traditional recommender systems are typically grouped into content based filtering and collaborative filtering. Each represents a different way of thinking about the problem. Content based filtering focuses on the basic characteristics of users and items and assumes that similarity in content reflects similarity in preference. In contrast, collaborative filtering relies on collective user-item interaction patterns and uses the hidden structure of historical feedback to infer recommendations without needing explicit content features (Su & Khoshgoftaar, 2009). Hybrid systems have emerged to address the limits of each approach by combining multiple signals to improve robustness and accuracy. With the rapid growth of mixed data, including social relationships, temporal activity, knowledge graphs and service invocation patterns, the field has increasingly moved towards graph based modelling. Graphs offer a flexible way to represent structured dependencies and higher order relationships that traditional matrix based methods struggle to encode. This shift has led to the use of GNNs, which provide a powerful framework for learning on graph structured data and have shown promising results across domains such as electronic commerce, social recommendation and service composition (Z. Wu et al., 2021).

Recent models such as NGCF(X. Wang, He, Wang et al., 2019) and LightGCN(X. He et al., 2020) demonstrate how graph message passing can boost recommendation quality by modelling higher order collaborative signals. The following sections review the foundations of the main recommendation paradigms and place this thesis within the growing intersection of recommender systems and graph based learning.

2.2.1 Content-Based Filtering

Content based filtering (CBF) is one of the earliest and most intuitive paradigms in recommender systems. Its core idea is that users tend to prefer items similar to those they have liked before. In practice, CBF models represent users and items with descriptive feature vectors derived from structured metadata, textual attributes or learned embeddings. Recommendations are produced by comparing a user profile, which is typically an aggregation of features from previously consumed items, with candidate items in the same feature space. This approach works well in domains with rich item descriptions such as documents, APIs, films and technical services. Considering it depends mainly on item content rather than collective user behaviour, CBF manages new or rare items effectively and often provides interpretable recommendations, for instance by identifying the attributes that drive similarity. Its performance is closely tied to the quality of item and user features, which motivates the adoption of advanced representation learning techniques in modern systems.

Background and Principles

CBF emerges from early information-retrieval research, where the aim is to represent documents and queries in a form that enables effective matching. In the vector space model of Salton, Wong and Yang (1975), the representation of documents as high-dimensional term vectors allows similarity estimation based on geometric comparison.

TF-IDF weighting becomes a standard technique for emphasising informative terms and is effective for automatic text retrieval (Salton & Buckley, 1988). Cosine similarity, which normalises vector length and handles sparse vectors well, becomes the dominant metric for comparing documents or user profiles with items.

Early recommender systems adapt these ideas by treating user histories as pseudo-documents constructed from the textual descriptions of items previously consumed. M. Pazzani and Billsus (1997); M. J. Pazzani and Billsus (2007) show that the use of simple vector space models together with incremental profile learning enables effective personalised news recommendation and establishes CBF as a practical paradigm.

However, TF-IDF and cosine similarity operate at the surface level of observable terms and struggle with synonymy, polysemy and deeper semantic relationships. This limitation encourages the development of latent feature learning. Latent semantic indexing (Deerwester, Dumais, Furnas, Landauer & Harshman, 1990) applies singular value decomposition to reduce dimensionality and uncover latent semantic factors, enabling more robust matching across different vocabularies. Topic modelling methods such as latent Dirichlet allocation (Blei, Ng & Jordan, 2003) further improve semantic representation by viewing documents as mixtures of interpretable latent topics. These topic-level features enhance CBF performance in domains with rich but noisy descriptions, including software services and APIs. Overall, CBF progresses from surface-level term weighting to increasingly sophisticated semantic representation learning.

User Representation

User representation is central to CBF because it forms the basis for modelling individual preferences. Traditional systems construct user profiles by aggregating feature representations of items that a user has consumed. Early work demonstrates that user profiles could be built from TF-IDF representations of previously viewed or preferred documents (M. Pazzani & Billsus, 1997; M. J. Pazzani & Billsus, 2007). Following

principles of information retrieval (Salton & Buckley, 1988), user histories are treated as pseudo-documents, which allows preference vectors to be estimated and similarity to candidate items to be computed directly. This approach forms the basis of many early text-rich content based recommenders.

Researchers later observe that preferences contain both stable and dynamic elements. Long-term interests reflect enduring tastes and change gradually over time. This motivates persistent user profiles that adapt slowly as users interact with more items (Billsus & Pazzani, 2000). Short-term interests represent immediate or situational needs shaped by recent browsing or queries. Session based studies, such as those of Shani, Brafman and Heckerman (2002) show the importance of modelling temporal dynamics to respond to rapidly shifting intentions. More recent systems integrate time decay functions or hybrid temporal models to capture both long-term stability and short-term responsiveness.

A major challenge for user representation is sparsity. Many users provide only limited interactions, resulting in incomplete or noisy profiles. Sparse histories weaken similarity estimation and often lead to the problem of unreliable recommendations Herlocker, Konstan, Borchers and Riedl (1999). In domains with complex or specialised content such as software services or APIs, sparsity can be even more problematic because users interact with only a small subset of available items. To mitigate this issue, research has explored profile enrichment, auxiliary information and hybrid frameworks (Burke, 2002), which compensate for limited data by incorporating additional signals. These developments highlight the importance of robust user representation for effective CBF.

Item Representation

Item representation is equally critical because recommendation quality depends on how well items are described. Traditional approaches rely on structured metadata, which

provide categorical or numerical attributes such as service type or functional category. For example, platforms like ProgrammableWeb list API categories, supported protocols and functional tags. Early recommender research shows that such metadata can be directly integrated into item profiles for similarity computation (Burke, 2002). Although metadata are interpretable and computationally efficient, they often oversimplify item characteristics and may be incomplete or inconsistently annotated.

To capture richer item characteristics, researchers increasingly turn to unstructured features drawn from textual descriptions, documentation, reviews or technical specifications. Text based features become influential when information-retrieval methods are applied to recommendation. TF-IDF representations (Salton & Buckley, 1988) support effective similarity computation and are widely used for items such as news articles (M. Pazzani & Billsus, 1997) and films (Mooney & Roy, 2000). Unstructured text is especially important where descriptions contain detailed functional or semantic information that metadata fail to reflect. However, vocabulary mismatch, noise and sparsity persist and they are the motivations for more expressive models.

Embedding-based item representations mark a major advancement. Neural embeddings such as word2vec (Mikolov, Chen, Corrado & Dean, 2013) and GloVe (Pennington, Socher & Manning, 2014) map textual attributes into dense vectors that reflect semantic relationships more effectively than bag-of-words approaches. These embeddings enable recommender systems to exploit distributional semantics for item modelling. Later developments introduce document embeddings (Le & Mikolov, 2014) and transformer-based models such as BERT (Devlin, Chang, Lee & Toutanova, 2019), which use contextualised language modelling to produce highly discriminative features. These methods are particularly effective in domains rich in unstructured text, such as API documentation and software tutorials. Embedding-based representations therefore provide more flexible and robust item modelling than traditional feature engineering.

Similarity Computation

Similarity computation underpins CBF by enabling comparison between user profiles and item representations. Traditional systems rely on classic similarity functions applied to vector-based content. Cosine similarity is widely used because of its suitability for high-dimensional sparse vectors that are common in TF-IDF models (Salton & Buckley, 1988). Other measures include the Euclidean and Manhattan distance, though they are less common in text settings because they are sensitive to vector magnitude. Jaccard similarity is used for discrete feature sets such as tags or keyword lists (Tan, Steinbach & Kumar, 2016). These classical measures remain popular due to their efficiency and interpretability.

As systems grew more complex, researchers identified limits in manually defined similarity functions and moved towards metric learning. Metric learning seeks to learn spaces in which semantically similar items are closer together. Early work such as the large-margin nearest neighbour method of Weinberger and Saul (2009) shows that learned Mahalanobis metrics can outperform hand-crafted ones. Neural approaches using Siamese networks and contrastive learning further advance this idea. Hadsell, Chopra and LeCun (2006) demonstrate that contrastive loss can produce embeddings that capture semantic similarity more accurately than traditional methods. Recent developments include BERT-based encoders and neural document embeddings fine-tuned with similarity objectives (Le & Mikolov, 2014; Devlin et al., 2019). Learned metrics capture non-linear and context-dependent similarity patterns, making them essential in domains where content relationships are complex.

2.2.2 Collaborative Filtering

Collaborative filtering represents a shift from content centred methods by using patterns in user and item interactions. Instead of relying on item metadata, collaborative filtering

infers preferences from behaviour observed across many users. The core assumption is that users who showed similar behaviour in the past are likely to share preferences in the future. This idea has produced some of the most influential recommendation algorithms, including neighbourhood methods, matrix factorisation and more recent neural collaborative filtering models.

Memory-Based Collaborative Filtering

Memory based collaborative filtering is one of the earliest and most widely deployed approaches for personalised recommendation. These methods compute predictions directly from the user and item interaction matrix without learning an explicit latent model(Su & Khoshgoftaar, 2009). Two principal variants dominate the literature. User-based collaborative filtering assumes that similar users express similar preferences, so the behaviour of neighbours is aggregated to estimate a target user's preference for an unseen item (Resnick et al., 1994). Item-based collaborative filtering identifies similarities between items by examining co-consumption patterns, and predicts a user's preference by assessing how the user rated similar items (Sarwar et al., 2001). Item based approaches have proven more scalable in commercial systems because item similarities tend to be more stable than rapidly changing user profiles.

Similarity computation is central to memory based collaborative filtering. Popular similarity metrics include Pearson correlation, which centres ratings to remove user level bias. Cosine similarity treats user or item vectors as points in a high dimensional space and measures angular distance. Jaccard similarity is used in implicit feedback settings where binary interactions such as clicks, purchases or service use are common (Herlocker et al., 1999). Each metric offers distinct advantages. Pearson correlation captures linear relationships in ratings. Cosine similarity is resilient to scale differences. Jaccard similarity is effective when explicit ratings or text based features are sparse.

Memory based methods face well documented scalability challenges. As the number

of users and items grows, computing pairwise similarities becomes computationally expensive. The time and space requirements increase quadratically with the number of users or items, which makes real time updates difficult in large systems (Linden, Smith & York, 2003). These methods also perform poorly when data are sparse because similarity estimates become unreliable when users share only a few co-rated items.

Model-Based Collaborative Filtering

Model-based collaborative filtering is developed to address the limitations of memory-based methods. The main idea is to learn latent representations of users and items so that the sparse interaction matrix can be transformed into dense lower dimensional factors that capture underlying preference structures (Koren et al., 2009). Matrix factorisation methods are the most influential within this class.

Singular value decomposition and related approaches decompose the user-item matrix into two low dimensional matrices that represent user and item latent factors. Probabilistic matrix factorisation extends this idea by introducing Gaussian priors over latent variables, which supports a fully probabilistic model that performs well with sparse data (Salakhutdinov & Mnih, 2007). Non-negative matrix factorisation constrains latent factors to be non negative and improves interpretability in domains where latent features represent additive attributes (D. Lee & Seung, 2000).

Training these models requires attention to regularisation and optimisation. Overfitting is a major concern because latent factors may memorise noise when data are sparse. L2 regularisation is commonly applied together with optimisation algorithms such as stochastic gradient descent or alternating least squares to ensure generalisation (Koren et al., 2009). Recent research also explores Bayesian priors and variational inference to manage uncertainty in parameter estimation.

Model-based collaborative filtering offers clear strengths. It scales to large datasets, captures complex user and item relationships through latent factors and often achieves

high predictive accuracy (Ricci, Rokach & Shapira, 2015). However, it also has limitations. Matrix factorisation requires a reasonable number of interactions for each user or item to learn meaningful embeddings. Classical models also have difficulty modelling non-linear behaviour and integrating side information such as text, images or contextual signals without significant modification.

Deep Learning Extensions

Deep learning has expanded the capabilities of collaborative filtering by enabling the modelling of non-linear relationships between users and items. Neural collaborative filtering (X. He, Liao et al., 2017) is one major contribution. It replaces the inner product interaction function of matrix factorisation with a multilayer neural network that can learn much more flexible match functions between user and item embeddings. This framework generalises classical methods and treats matrix factorisation as a special case.

Autoencoder based architectures provide another important extension. Denoising autoencoders and variational autoencoders have been adapted for recommendation, particularly in implicit feedback settings where models must reconstruct binary interaction matrices (Sedhain, Menon, Sanner & Xie, 2015). These approaches encode user behaviour into dense representations and predict missing interactions by reconstructing partially corrupted user vectors. They also handle sparsity more effectively than linear factorisation models because deep neural networks can infer latent structure even when explicit signals are limited.

Deep learning based collaborative filtering therefore increases flexibility and accuracy, although it introduces greater computational demands and requires large training datasets.

Cold Start and Sparsity Problems

Collaborative filtering methods rely on interaction data. When new users or items enter the system or when the interaction matrix is extremely sparse, predictions become unreliable. This is known as the cold start problem (Lam, Vu, Le & Duong, 2008). Sparsity also reduces the quality of similarity estimates and latent factor learning.

Several strategies have been proposed to address these challenges. One common approach is hybridisation, where collaborative signals are combined with content based features such as textual descriptions, metadata or contextual information (Burke, 2007). Hybrid recommenders use side information to initialise user or item representations, which enables recommendations even with limited interaction data.

Another important direction is transfer learning. Knowledge learned in one domain, such as film ratings, can be transferred to another related domain, such as book recommendations, by using shared embeddings, multi-task networks or cross domain alignment (S. J. Pan & Yang, 2010). Transfer learning is particularly valuable when data are scarce because it allows models to leverage external datasets and produce richer latent representations.

2.3 Mashup-API Recommendation

Mashup-oriented API recommendation has emerged as a significant research area as modern software ecosystems shift toward service modularity, reuse, and composition. In these environments, APIs act as independent yet combinable functional units, while mashups constitute higher-level applications that integrate multiple APIs to achieve complex objectives. Public repositories such as ProgrammableWeb, with tens of thousands of APIs and thousands of mashups, provide a rich landscape of interaction histories, textual documentation, invocation semantics and contextual metadata. This ecosystem

supports large-scale analyses of how APIs relate to one another, how they are combined in practice, and how developers express requirements. Earlier research highlighted the potential of mashup logs and API documentation for recommendation, but the field has since expanded substantially by exploring deeper semantic, behavioural, and structural signals. A recent survey (Alhosaini, Alharbi, Wang & Xu, 2024) emphasises that the increasing scale and heterogeneity of API ecosystems, combined with evolving developer needs, call for sophisticated recommender models capable of leveraging diverse data modalities and reasoning across relational structures.

2.3.1 The API-Mashup Ecosystem and Its Information Modalities

The expanding ecosystem of APIs and mashups forms an information-rich environment that provides natural opportunities for recommendation. Early analyses of API collections describe ecosystems in which functional overlap, category membership, and textual metadata provided basic signals of API relatedness. Xia et al's clustering-based studies (B. Xia et al., 2015), for instance, reveals that APIs naturally group into functional clusters based on descriptive and behavioural features, suggesting that clustering could guide mashup development and support recommendation tasks.

The evolution of mashup tools and platforms has also stimulated research into automation. Q. Xue, Liu, Chen and Chuah (2017) propose automatic mashup generation driven by natural-language descriptions, demonstrating that requirement understanding could be automated using NLP pipelines even before neural models became widespread. As ecosystems grow, more sophisticated representations become necessary. Knowledge graph perspectives have increasingly influenced research, with several works showing that real-world API environments contain relational structures linking APIs, mashups, tags, categories, and co-invocation patterns. X. Wang, Liu, Liu, Chen and Wu (2021) construct such a knowledge graph to support unsupervised API recommendation and

demonstrate that effectively capturing entity relationships and semantic neighbourhoods significantly enhances recommendation performance without requiring labelled data.

Another important trend has been the recognition that developer requirements are complex and often expressed in free text. Sang, Deng and Liao (2023) address this challenge by applying adversarial semantic modelling to extract deep semantic features from textual requirements. As a result, the models become capable of capturing rich intent signals. This change toward understanding textual requirements is similar to frameworks such as MTFM (H. Wu, Duan, Yue & Zhang, 2022), which integrates refined text encoders with metadata-aware components. Together, these studies demonstrate that modern API ecosystems contain heterogeneous data sources such as text, network structure, categories, and quality-of-service metadata that must be combined for effective mashup-oriented API recommendation.

2.3.2 Challenges in Mashup-Oriented API Recommendation

A central challenge in mashup-API recommendation arises from the fact that APIs are not independent items. Compositions demand a deep awareness of functional dependencies and workflow compatibility. Mashups frequently rely on multi-step functional pipelines, such as geolocation followed by map rendering and then route computation, where each step imposes constraints on the data produced by the previous one. Early research highlights that classical collaborative filtering could not account for these compositional dependencies (K. Huang, Fan & Tan, 2014), and subsequent studies confirm this limitation. Qi et al. (2019) explicitly model compatibility through a weighted correlation graph designed to verify whether candidate APIs can work together within a keyword-driven mashup query. Building on the same insight, H. Wu et al. (2022) incorporates a compatibility metric into their multi-task learning framework and shows that content-based relevance alone is insufficient when APIs conflict due

to protocol mismatches, divergent authentication mechanisms or incompatible data schemas. The limitations of simple co-invocation statistics are further highlighted by research on random-walk-based knowledge graphs (X. Wang et al., 2021) and implicit co-invocation modelling (Yao, Wang, Sheng, Benatallah & Huang, 2021), both demonstrating that compatibility must be inferred from higher-order connections and indirect relational paths rather than simple co-occurrence.

Another long-standing challenge are the semantic ambiguity and diversity of developer requirements. Traditional topic-model-based approaches struggle to represent the complexity of natural-language descriptions, prompting a shift toward deep semantic modelling. Studies such as Sang et al. (2023) show that full-text requirement understanding demands models that can capture both local and global semantics. Earlier efforts like Cao et al. (2020), which integrates content with network-based clustering, illustrate that combining multiple sources of semantics that improves alignment between requirements and API capabilities. Similarly, keyword-search models augmented by graph neural networks, such as KS-GNN (G. Kang, Wang et al., 2024), demonstrate that mapping textual requirements to structural semantic representations can substantially improve API retrieval.

Data sparsity also remains a pervasive issue in Mashup API ecosystems. Most mashups use only a handful of APIs, and the majority of APIs appear only a few times in historical logs. This long-tail distribution undermines models that rely heavily on co-invocation or historical similarity. Popularity bias, which causes recommenders to overemphasise frequently used APIs while neglecting specialised but relevant ones, is addressed in several studies. Zhai et al. (2025) propose explicit de-biasing mechanisms for multimedia API recommendation and demonstrate that mitigating popularity effects significantly improves the recommendation of rarely used services. Graph-based collaborative filtering methods, such as neural graph collaborative filtering (Lian & Tang, 2022), heterogeneous information networks (M. Tang, Xie, Lian, Mai & Li, 2024), and

factorisation machines (G. Kang, Liu, Cao & Cao, 2020; G. Kang, Liang et al., 2024), attempt to alleviate sparsity by leveraging relational structures that allow information to propagate across nodes. However, even these advanced techniques struggle in extreme sparsity scenarios where both mashups and APIs have minimal historical interactions.

A final challenge concerns the dynamic nature of mashup development. As new APIs emerge and old ones evolve, recommendation models must adapt to changing contexts. Reinforcement learning-based approaches (Anarfi & Fletcher, 2019) introduce adaptivity, which allows systems to refine their predictions based on developer interactions, feedback, and sequential decision-making processes. Active recommendation models based on usage history attempt to adjust recommendations dynamically. Despite these advances, most recommendation systems remain static and few account for temporal evolution or interactive learning.

2.3.3 Existing Approaches

Several methodological families have emerged to address these challenges, each offering complementary strengths and limitations. Content-based approaches form the earliest lineage of research. Initial models relied on lexical similarity, TF-IDF representations, and LDA topic models to match API descriptions to requirements (Q. Huang, Xia, Xing, Lo & Wang, 2018). These methods provide a baseline understanding of semantic relevance but were limited by their inability to capture deeper functional meaning. Subsequent efforts introduce more sophisticated semantic modelling. Full-text mining approaches such as those proposed by Sang et al. (2023) employ adversarial learning and deep neural encoders to extract semantics from lengthy requirement documents. Neural semantic matching approaches (T. Yu, Liu, Zhang & Liu, 2023) incorporate convolutional or recurrent neural networks to represent textual API descriptions, reflecting

local context and global meaning. Multi-level semantic structures, including tag-topic-co-occurrence networks (H. Li et al., 2017) and content-network hybrid clustering (Cao et al., 2020), further demonstrate that integrating textual features with metadata or community structure improves semantic coherence.

Collaborative filtering constitutes another foundational strand. Classical user-based and item-based CF (Z. Zheng & Lyu, 2013) provides initial insight into preference modelling within mashup ecosystems. Matrix factorisation methods, particularly probabilistic MF approaches (Yao et al., 2021), extended this by learning latent structures underlying mashup-API associations. Personalised hybrid CF frameworks (Y. Jiang, Liu, Tang & Liu, 2011) introduce personalised influence modelling, which enables more accurate similarity estimation. However, reliance on historical invocations limited these models in sparse settings. Later works incorporate auxiliary signals, such as topics and user interaction features (M. M. Rahman & Liu, 2020), to enhance matrix factorisation. Neural graph collaborative filtering (Lian & Tang, 2022) leverages message passing to capture high-order correlations and demonstrates that graph-enhanced CF can significantly outperform shallow models.

Hybrid clustering, graph mining, and network-based approaches seek to capture structural relationships between APIs and mashups. Clustering-based studies (B. Xia et al., 2015) reveal that APIs often form coherent functional groups that align with developer's needs. Network-based clustering and integrated MF methods (Cao et al., 2020) use network semantics to improve recommendation accuracy. Manifold-learning techniques (W. Gao, Chen, Wu & Gao, 2015) attempt to capture non-linear similarity structures through graph-based embeddings, showing that geometric relationships among APIs can be highly predictive of functional relevance.

With the growing recognition that API ecosystems contain complex relational structures, knowledge graph and heterogeneous information network approaches have become increasingly popular. Random-walk-based knowledge graph embedding methods

(X. Wang, Wu & Hsu, 2019) treat co-invocation patterns and category relationships as nodes and edges in a graph, learning embeddings through truncated random walks. Early random-walk models (X. Wang et al., 2021) demonstrate the viability of knowledge graphs for capturing both functional similarity and long-range structural dependencies. More recently, M. Tang et al. (2024) introduces a pre-trained HIN model, showing that meta-path reasoning across entities such as APIs, mashups, tags, and categories significantly enhances semantic alignment and recommendation performance. Keyword-search GNN frameworks such as KS-GNN (G. Kang, Wang et al., 2024) similarly use graph structure to enrich requirement understanding. Although less prevalent in mashup contexts, trust-aware recommendation models (G. Yang, Yang & Jin, 2021) reveal that social trust structures can also enhance recommendation accuracy and suggest potential for incorporating social knowledge graphs into API recommendation frameworks.

A final methodological strand centres on advanced deep learning, multi-model fusion, and multi-task learning. Factorisation-machine-based frameworks, including NAFM and SANFM (G. Kang et al., 2020; G. Kang, Ding, Liu, Cao & Xu, 2023), model higher-order interactions among content features, metadata, and network structure. These models demonstrate that capturing non-linear interactions through neural feature mapping significantly enhances performance in sparse environments. Multi-model fusion frameworks, particularly MTFM and MTFM++ (H. Wu et al., 2022), combine semantic encoders, feature interaction modules, and auxiliary tasks such as category prediction to yield comprehensive representations of mashups and APIs. Reinforcement learning models (Anarfi & Fletcher, 2019) introduce interactivity and temporal adaptivity while active recommendation methods based on usage history provide mechanisms for dynamic refinement.

2.3.4 Limitations and Research Gaps

Despite substantial advancements, several limitations persist across the literature. Many models still underutilize the full relational richness of the API-mashup ecosystem. Although graph-based methods have improved structural reasoning, most limit themselves to simplified bipartite or homogeneous graphs and fails to incorporate multi-relation heterogeneity involving tags, providers, categories, semantic clusters, and workflow dependencies. Even sophisticated models such as HINs and KG-embedding frameworks tend to isolate structural features from textual semantics, resulting in suboptimal integration. There is also a persistent gap in modelling functional workflows. Compatibility-aware methods exist, but comprehensive modelling of input-output relationships, data-flow constraints, and sequential dependencies remains rare. This omission restricts the practical utility of recommendations, as semantically similar APIs may be functionally incompatible. Semantic and structural fusion also remains incomplete. Although deep neural text encoders and GNNs have advanced separately, few models integrate these modalities in a cohesive end-to-end architecture. Recent contrastive learning approaches suggest promising directions, but their scope remains limited.

Sparsity and long-tail distributions continue to hinder model performance. Even models designed to mitigate popularity bias struggle with extremely sparse APIs that appear only once or twice in historical logs. Furthermore, temporal and interactive modelling is limited. Reinforcement learning approaches demonstrate that adaptivity is critical for real-world applicability, yet few models consider the dynamic evolution of APIs and mashups. Finally, current research still lacks holistic frameworks that jointly model content, structure, compatibility, temporal dynamics, and developer interaction. Consequently, there is substantial scope for integrative models that can address the complex nature of mashup-oriented API recommendation.

2.4 Network Motifs

Network motifs are recurring and statistically significant subgraph patterns that reflect fundamental building blocks of complex networks. In recommendation contexts, motifs offer a way to understand higher order relationships that extend beyond simple pairwise interactions. For example, they can capture clusters of mashups that invoke similar APIs or triadic structures that reflect complementary service functionality. Incorporating motif information enables models to recognise local structural patterns that provide informative features for machine learning methods, particularly within graph based recommender systems.

2.4.1 Definition and Historical Development

The formal definition of network motifs originates from efforts to understand organisational patterns that cannot be captured through global measures such as degree distributions or shortest path length. Milo et al. (2002) provides the first rigorous definition of a network motif as:

A subgraph H is a network motif of graph given the number of occurrences of H in G is significantly higher than in an ensemble of random graphs that preserve key structural properties of G, typically the degree sequence.

This formulation introduces a clear statistical framework in which candidate subgraphs are counted in the observed network, compared with an ensemble of random graphs, and designated as motifs when their over-representation cannot be attributed to chance.

Later work refines both the conceptual and methodological basis of motif analysis. Stone et al. (2019) review foundational aspects and emphasise that the choice of the null model is critical since different constraints such as degree sequence, reciprocity or community structure can lead to different sets of motifs. Over time, motif research

expands from directed triads in relatively small biological systems to larger subgraphs and to networks involving multiple node or edge types, which reflects the growing complexity of empirical datasets.

2.4.2 Motifs in Social, Biological, and Information Networks

Motif analysis was first applied extensively in systems biology. In transcriptional regulatory networks and metabolic pathways, motifs such as feed forward loops, feedback loops and bi-fan structures were found to perform basic information processing functions, including signal filtering, response acceleration and noise reduction (Milo et al., 2002; U. Alon, 2007). These findings encourage the interpretation of motifs as elementary computational circuits shaped through evolution to carry out specific tasks.

Similar patterns have been observed in social networks. Triadic closure, which reflects the likelihood that two individuals with a shared acquaintance will also form a link, appears as an overrepresented triangle motif in many empirical graphs (Holland & Leinhardt, 1971; Granovetter, 1973). Motifs associated with reciprocity or transitivity in directed social networks have been linked to processes such as trust formation, collaboration and the spread of information (Wasserman & Faust, 1994; M. E. J. Newman, 2003; Easley, Kleinberg et al., 2010).

Information and technological networks also exhibit characteristic motifs. Citation networks reveal recurrent triads that correspond to scholarly influence and the formation of topical communities. The World Wide Web demonstrates motifs related to navigation structures and mutual linking between pages. Software dependency networks and electronic circuits contain motifs that correspond to reusable modules or logic gates. Collectively, these findings support the idea that motif frequency distributions act as structural signatures that reveal the generative and functional processes underlying different classes of networks (Milo et al., 2002; Stone et al., 2019).

2.4.3 Motif Detection Algorithms

The growing use of motif based analysis has motivated the development of efficient algorithms for motif detection. Early methods relied on exhaustive enumeration of all subgraphs of a given size together with subgraph isomorphism checks, a procedure that becomes computationally expensive even for moderately sized networks. Wernicke and Rasche (2006) address this challenge by introducing the FANMOD tool, which combines an efficient enumeration strategy with random sampling to detect motifs in larger graphs and for larger motif sizes.

More recent work has focused on scalability and flexibility. Sampling-based techniques and colour-coding methods reduce computational cost while still yielding statistically meaningful subgraph count estimates (N. Alon, Yuster & Zwick, 1995; Kashtan, Itzkovitz, Milo & Alon, 2004; Wernicke & Rasche, 2006). Parallel and distributed implementations including MapReduce-style frameworks and GPU-accelerated systems allow motif analysis to be applied to graphs with millions or billions of edges (Y. Liu, Schmidt, Liu & Maskell, 2010; M. Rahman, Bhuiyan & Al Hasan, 2014; W. Lin, Xiao, Xie & Li, 2016). As interest in heterogeneous information networks has grown, algorithmic frameworks now extend motifs to incorporate node and edge types. This development yields typed graphlets and heterogeneous motifs that capture richer semantics in domains such as knowledge graphs and multi-relational recommender systems (Shervashidze, Vishwanathan, Petri, Mehlhorn & Borgwardt, 2009; Soundarajan, Eliassi-Rad & Gallagher, 2014; R. A. Rossi et al., 2019).

2.4.4 Motifs for Modelling Higher-order Dependencies

Motifs have also been used to model higher order dependencies that cannot be represented by pairwise edges alone. Classical network representations assume that all interactions can be decomposed into dyads, an assumption that fails in many real

systems where meaningful interactions involve three or more entities simultaneously. Benson et al. (2016) propose a general framework that clusters networks according to their higher order connectivity patterns, replacing edges in the clustering objective with motif occurrences. This work demonstrated that communities derived from motifs can differ markedly from those identified using standard edge based methods, revealing organisational principles that remain hidden at the pairwise level.

In representation learning, researchers have incorporated motif structure into graph neural architectures. Monti, Otness and Bronstein (2018) develope motif based graph convolutional and attention networks in which the message passing process is guided by motif induced adjacency matrices or motif weighted neighbourhoods rather than simple edge connectivity. These models showed that higher order information can significantly improve performance on semi supervised classification and link prediction tasks, particularly in directed or heterogeneous graphs. By focusing on subgraphs instead of individual edges, motif based models can capture co usage patterns, functional chains and other multi step dependencies that pairwise statistics alone cannot reveal. This capability is especially valuable in recommendation scenarios where user behaviour and item relationships frequently depend on interactions involving multiple entities or attributes.

2.4.5 Applications in Recommender Systems

Motif based methods have only recently begun to appear in recommender system research, yet the literature is expanding rapidly. H. Zhao, Zhou, Song and Lee (2019) introduce motif enhanced recommendation for heterogeneous information networks and argued that metapath based approaches mainly capture first order relations while ignoring higher order interactions among nodes of the same type. Their model constructs motif based similarity measures that better leverage co occurrence patterns

among users and items. Subsequent work has incorporated motif structure directly into graph neural recommenders. Motif based graph attention networks for service recommendation aggregate information not only from immediate neighbours but also from motif defined regions of the graph, thereby exploiting higher order context in service and API networks (G. Wang et al., 2023). Motif guided graph convolutional networks for collaborative filtering construct motif specific adjacency matrices that drive neighbourhood aggregation in bipartite user item graphs and have demonstrated improved performance over strong baselines in benchmark evaluations (Y. Zhang et al., 2025). Recent attention based models, including MGAT and MGATv2, further integrate motif information with attention mechanisms and achieve substantial performance gains by better capturing higher order structural features (Sheng et al., 2024).

Across these studies, motifs have been used to define more expressive similarity measures to augment input features with motif statistics and to construct higher order neighbourhoods for graph neural networks. They also offer potential for improving interpretability, since the recommendations produced by motif based models can be linked to identifiable structural patterns in the graph.

2.5 Graph Neural Networks

Graphs are a type of mathematical abstract structure that are typically used to represent network-structured data, such as social networks, the World Wide Web, the Internet of Things, molecular networks, and road networks. Composed of nodes and edges, graphs are naturally capable of capturing interactions between entities. This makes graphs a popular choice for modelling data especially in modern deep learning tasks. For example, it is desirable to learn feature embeddings to predict people's roles in collaborative networks, to detect certain communities in social media, to provide recommendation on online shopping websites, or to learn molecular representation for

drug discovery.

The key challenge of graph representation learning is that graphs are in non-Euclidean space, which explains why existing Convolutional Neural Networks (CNNs) designed for image data are not suitable for graph data. Recently, numerous graph neural networks (GNNs) have been proposed to better learn representations on graphs. GNNs show great performance on graph learning tasks including node classification, graph classification, and link prediction. As a result, GNNs have been successfully applied in multiple industries such as Internet of Things (W. Zhang et al., 2019; S. Liu et al., 2020; Protogerou, Papadopoulos, Drosou, Tzovaras & Refanidis, 2021), e-Commerce (Piao, Zhang, Xu, Chen & Li, 2021; Z. Wang et al., 2020; W. Yu & Qin, 2020; X. Zhou et al., 2019) and social media (Anwaar et al., 2021; X. He et al., 2020; Q. Wu et al., 2019).

In this section, we mainly consider undirected graph, which is denote as $G = (V, E)$ where V is the set of nodes and E is the set of edges. N is the number of nodes in G. The adjacency matrix of G is represented as $A \in \mathbb{R}^{N \times N}$ where the element $A_{ij} = 1$ if nodes i, j are connected and $A_{ij} = 0$ if nodes i, j are not connected. A graph's node feature matrix is represented as $X \in \mathbb{R}^{N \times d}$ with $x_i \in \mathbb{R}^d$ denoting a single node i's feature vector. The neighbours of node i are represented as $\mathcal{N}(i) = \{j | A_{ij} = 1\}$. Superscripts are used along with matrices and vectors to denote the variables' current layer. For example, H^l is the representation of the l^{th} hidden layer and h_i^l is the representation of node i at the l^{th} hidden layer. The sigmoid activation function is denoted by $\sigma(\cdot)$. The following table summarises the notations used in this paper.

Here we present a simple taxonomy of GNNs. We divide GNNs into four categories: Recurrent GNNs, spectral GNNs, spatial GNNs, and topology-aware GNNs. Recurrent GNNs are the first type of GNNs. Inspired by the architectures of recurrent neural networks, recurrent GNNs recursively update nodes' features from their neighbours until convergence. Spectral GNNs redefine the convolution operation to adapt graph data. They typically extend the notion of Fourier transform or its extensions to graph

Table 2.1: Notations used in this section.

Notation	Description
G	A graph
V	The set of nodes in a graph
E	The set of edges in a graph
A	The adjacency matrix of a graph
X	The feature matrix of a graph
x_i	The feature vector of node i
$H^{(l)}$	The feature matrix at layer l
$h_i^{(l)}$	The feature vector of node i at layer l
$\sigma(\cdot)$	The sigmoid activation function

domain. Spatial GNNs redefine the convolution operation by considering the kernel behaviour of convolutional neural networks. Topology-aware GNNs explore local and global topological structures on graphs using techniques such as Weisfeiler-Lehman isomorphism test or incorporating network motifs into graph filter design.

2.5.1 Ground-Breaking GNNs

The vanilla GNN (Scarselli, Gori, Tsoi, Hagenbuchner & Monfardini, 2009) is the first properly designed graph neural network model. The objective of GNN is to learn a hidden state embedding h_v of each node v by collecting information from the node's neighbours in an iterative manner. This iterative information aggregation process can be described by (2.1):

$$h_v = f\big(x_v, e_v, h_{\mathcal{N}(v)}, x_{\mathcal{N}(v)}\big) \tag{2.1}$$

where $x_v, e_v, h_{\mathcal{N}(v)}, x_{\mathcal{N}(v)}$ are the features of node v, features of edges of v, hidden states of v's neighbours, features of v's neighbours respectively. The function f is the local transition function that is shared by every node in the graph. This process is also known as message passing (Gilmer et al., 2017) and is later widely used by many spatial

GNNs.

After obtaining converged hidden states, the output is calculated using (2.2):

$$o_v = g(h_v, x_v) \tag{2.2}$$

where g is the local output function.

(2.1) and (2.2) can be rewritten as:

$$H = F(H, X) \tag{2.3}$$

$$O = G(H, X_N) \tag{2.4}$$

where H, O, X and X_N are matrices that stack all hidden states, outputs, features, and node features respectively. F and G are stacked versions of f and g.

The iterative information aggregation needs a constraint to converge. Here GNN adopts Banach's fixed point theorem which requires the global transition function F to be a contraction map.

The main limitations of GNN derive from its iterative structure. First of all, GNN is computationally expensive since each hidden state needs to be updated recursively to reach convergence. Moreover, the constraint of convergence, i.e., Banach's fixed point theorem, leads to the problem of over-smoothing. This limitation makes GNN undesirable for node representation learning.

From the structure of GNNs, we can easily find some similarities between GNNs and RNNs. In fact, there are several follow-ups to continue exploring the idea of recurrent GNNs. Gated Graph Sequence Neural Network(GGSNN) (Y. Li, Tarlow, Brockschmidt & Zemel, 2017) uses a gated recurrent unit (GRU) to update the hidden

state instead of a contraction map function:

$$h_v^{t+1} = GRU(h_v^t, \sum_{u \in \mathcal{N}(v)} (W_e h_u^t)) \tag{2.5}$$

GGSNN improves against GNN in several aspects. Firstly, the update of hidden states does not need to converge so the steps of an update are limited to a fixed number. Secondly, the nodes do not have a constraint of parameters for convergence. Lastly, GGSNN has different learnable parameters for different edge types.

Inspired by the success of Convolutional Neural Networks, researchers have re-defined convolution in the context of graph data. Graph convolutions are constructed based on the spectral graph theory, where the name of Spectral GNNs derives from. As the first type of spectral GNN introduced, Spectral CNN (Bruna et al., 2014) is an attempt of generalising CNNs into graph domain. The convolution operation of CNNs is defined in Euclidean space. The convolution kernel of CNNs has a fixed size since each node has a fixed number of neighbours in Euclidean space. However, graph data are defined in non-Euclidean space, which means the number of neighbours for each node may be different. To solve this problem, the authors extend the notion of convolution via an analogical reasoning from Laplacian and Fourier basis to graph Laplacian and its eigenvector:

$$L = I_N - D^{-1/2} A D^{-1/2} = U \Lambda U^T \tag{2.6}$$

where L is the normalised graph Laplacian, D and A are degree matrix and adjacency matrix respectively, U is a matrix consisting of eigenvectors of L, Λ is a diagonal matrix of L's eigenvalues. Then the convolution operation on graphs is defined as:

$$x * g = U(U^T x \odot U^T g) \tag{2.7}$$

where $\odot$ denotes element-wise multiplication. This formula can be simplified by

denoting $U^T g$ as a kernel g_θ:

$$x * g_\theta = U g_\theta U^T x \tag{2.8}$$

Then the key is to compute the kernel g_θ.

For spectral CNN, the kernel is a diagonal matrix of N learnable parameters. Due to the high computational cost of eigen-decomposition, spectral CNN has a complexity of $O(N^3)$. To improve the efficiency of spectral CNN, ChebNet (Defferrard et al., 2016) uses Chebyshev polynomials to approximate the kernel:

$$x * g_\theta = \sum_{k=0}^{K-1} \theta_k T_k(\widetilde{L}) x \tag{2.9}$$

where $\widetilde{L} = 2L/\lambda_{max} - I_N, T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$, and $T_0(x) = 1, T_1(x) = x$.

GCN (T. N. Kipf & Welling, 2017) makes a first-order approximation of ChebNet by assuming $K = 1$ and $\lambda_m ax = 2$:

$$x * g_\theta = \theta_0 x - \theta_1 D^{-1/2} A D^{-1/2} x \tag{2.10}$$

To further reduce the number of parameters, GCN assumes $\theta_0 = -\theta_1 = \theta$.

Now we have:

$$x * g_\theta = \theta(I_N + D^{-1/2} A D^{-1/2}) x \tag{2.11}$$

The layer-wise propagation rule is described by:

$$H^{l+1} = \sigma(D^{-1/2} A D^{-1/2} H^l W^l) \tag{2.12}$$

Through the approximation, ChebNet and GCN can reach linear complexity with respect to the number of edges.

Several other spectral GNNs are proposed to improve performance. CayleyNets

(Levie, Monti, Bresson & Bronstein, 2019) applies Cayley polynomials to include narrow frequency bands, with no efficiency degradation compared to ChebNet. Adaptive graph convolutional network (AGCN) (R. Li, Wang, Zhu & Huang, 2018) learns unique residual graph Laplacian for each sample in batch by learning distance metric. In this way, AGCN can take graphs of arbitrary structures as input. Dual Graph Convolutional Networks (DGCN) (Zhuang & Ma, 2018) learns local and global structural information concurrently in two convolutional neural networks. Graph Wavelet Neural Network (GWNN) (B. Xu, Shen, Cao, Qiu & Cheng, 2018) adopts graph wavelet transform, which does not require eigen-decomposition. This enables GWNN to have a low computational complexity of linear with respect to the number of edges.

As the first model of spectral GNNs, Spectral CNN brings the concept of convolution from Euclidean space to graph domain by finding graph Laplacian as the counterpart of Fourier transform in Euclidean space. ChebNet, GCN, and CayleyNet are all follow-up works of Spectral CNN that improve computing efficiency from approximation.

Spatial GNNs define convolution from another perspective. Conventional CNNs use a convolution kernel to update current pixel's value by calculating the weighted average of surrounding pixels' values. Similarly, spatial graph convolution is defined as the process of updating current node's representation by aggregating the representations of neighbouring nodes. A simple example of Spatial GNNs is GCN. Although we have mentioned GCN as a spectral GNN above, GCN can be rewritten in a spatial form:

$$h_i^{l+1} = \sigma(\sum_{j \in N_i} \frac{1}{c_{ij}} h_j^l W^l) \tag{2.13}$$

where $c_{ij} = \sqrt{d_i d_j}$. The term d_i denotes the degree of node i.

The connection between (2.12) and (2.13) can be easily understood by considering the initiative of matrix multiplication in (2.12). After removing the weights matrix and the matrices for symmetry and normalisation, we derive the adjacency matrix A and

the feature matrix H. The multiplication of A and H is simply the aggregation of the features for each node's neighbours which is consistent with (2.13).

GraphSAGE (W. Hamilton et al., 2017) also adopts the idea of aggregating information from neighbours. It samples from neighbours to improve efficiency and avoid overfitting. Apart from sum-pooling, their model can use other aggregation methods including max-pooling and long short-term memory (LSTM). The key advantage of GraphSAGE over GCN is the batch-training algorithm, which allows GraphSAGE to avoid using the entire graph as input for training, making it known for its inductive learning ability.

Gaussian Mixture Model Network (MoNet) (Monti, Bronstein & Bresson, 2017) defines pseudo-coordinates for node pairs to represent the relationship between nodes and their neighbours. For graphs, pseudo-coordinates are constructed using node degrees:

$$u(i,j) = \tanh(A(\frac{1}{\sqrt{deg(i)}}, \frac{1}{\sqrt{deg(j)}})^T + b) \tag{2.14}$$

Pseudo-coordinates are used in MoNet to construct the weight function for each node pair:

$$w_k(u) = \exp(-\frac{1}{2}(u-\mu_k)^T \Sigma_k^{-1}(u-\mu_k)) \tag{2.15}$$

where Σ_k is the learnable covariance matrix of a Gaussian kernel k and μ_k is the mean vector of the Gaussian kernel. The propagation rule of MoNet can be written as:

$$h_i = \sigma(\sum_{k=1}^{K} \sum_{j \in N_i} w_k(u) U^k h_j) \tag{2.16}$$

where K is the number of kernels.

Graph Attention Network (GAT) (Veličković et al., 2018) adopts the attention

mechanism which is provably powerful in application of natural language processing (NLP) (Bahdanau, Cho & Bengio, 2016; Vaswani et al., 2017). The attention weight essentially measures the similarity between each pair of connected nodes. It is calculated between each node and its neighbours during convolution before used for weighted neighbourhood aggregation. The feature update propagation of GAT is given by (2.17):

$$h_i^{l+1} = \sum_{j \in \mathcal{N}(i)} \alpha_{i,j} W^l h_j^l \tag{2.17}$$

where $\alpha_{i,j}$ denotes the attention weight between nodes i and j.

$$\alpha_{ij}^l = \mathrm{softmax_i}(e_{ij}^l) \tag{2.18}$$

$$e_{ij}^l = \mathrm{LeakyReLU}\left(\vec{a}^T [W h_i \| W h_j]\right) \tag{2.19}$$

Spectral Attention Network (Kreuzer, Beaini, Hamilton, Létourneau & Tossou, 2021) proposes to enhance Transformer (Vaswani et al., 2017) on graphs by incorporating learned positional encodings to node embeddings. The learned positional encoding is generated from a selected number of lowest-eigenvalue eigenvectors of the graph Laplacian.

Geometric Graph Convolutional Networks (Pei et al., 2019) transforms the graph into a latent space where aggregation is determined by the geometric relationship between neighbours. It maps the original graph to a latent continuous space where the neighbours of each node are different from those in the original graph. The aggregation of neighbourhood information obeys a geometric relationship.

MixHop (Abu-El-Haija et al., 2019) is a higher-order message passing model that can learn Delta Operator, which is not learnable for GCN. It shows the ability of learning a wider class of representations such as Gabor-like filters with no compromise on computational and memory complexity compared to GCN. Specifically, it repeatedly

concatenates powers of adjacency matrix in each layer. Since multiple powers of adjacency matrices are used, the searching space is significantly larger than vanilla GCNs. To solve this, the authors use a lasso regularization to automatically learn the model architecture.

The inductive bias of general GNNs can be summarised as knowledge migration from other research fields. For Recurrent GNNs, the inductive bias is essentially about methods that determine when to stop iteration of RNNs. For spectral GNNs, it is the convolution operator in Euclidean space. An example for spatial GNNs is GAT where the attention mechanism is adopted from NLP.

A group of recent works show that the topological structure of graphs has significant influence on local and global representation learning. Most of these works can be categorised as spatial GNNs but we focus on the distinction of their network structures and hope to extract more insights and inspiration behind these ideas. Here we further divide topology-aware GNNs into two sub-categories according to how topological information is involved in their model design: WL-test-based GNNs and motif-based GNNs.

2.5.2 Topology-Aware GNNs

Despite the abundance of high-performing GNNs, many are solely empirical and lack support from theoretical analyses. In this section, we discuss how Weisfeiler-Lehman isomorphism test (WL test) (Weisfeiler & Leman, 1968) is incorporated into graph representation learning. WL test is proposed to solve graph isomorphism problems in polynomial time. Despite some counter-examples found later, WL test is still capable for most graphs in the probabilistic sense (Babai, Erdös & Selkow, 1980).

A series of works focus on revisiting GNNs from a graph theory perspective. The authors of Graph Isomorphism Network (GIN) (K. Xu et al., 2019) show that any

message-passing GNNs are at most as powerful as the WL test in terms of distinguishing non-isomorphic graphs. (2.20) shows a general structure of updating node representations for a message-passing GNN. To achieve the powerfulness as described in the paper, the update function Φ and message function f, and the graph-level readout function must be injective.

$$h_v^t = \Phi\Big(h_v^{t-1}, f\big(\{h_u^{t-1} | u \in \mathcal{N}(v)\}\big)\Big) \tag{2.20}$$

GIN is then presented as a simple example of powerful GNNs that generalises the WL test:

$$h_v^t = MLP^t\Big((1+\epsilon^t)h_v^{t-1} + \sum_{u \in \mathcal{N}(v)} h_u^{t-1}\Big) \tag{2.21}$$

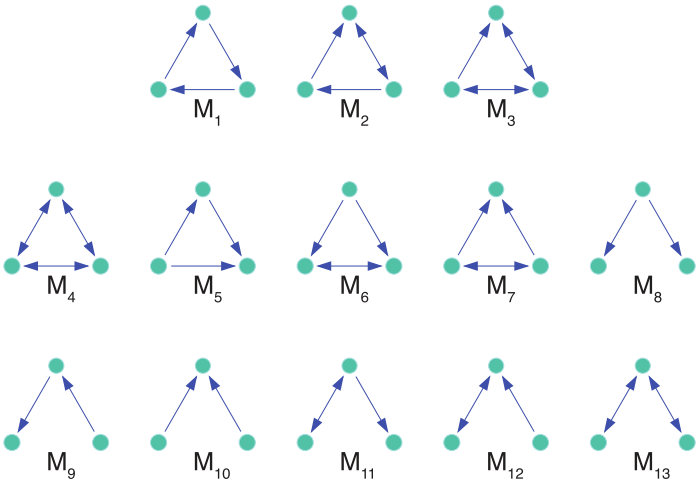
where MLP and the learnable parameter ϵ are used to learn injective functions.

GIN is powerful but has the same drawbacks as the WL test such as the absence of the ability to capture graph theoretical properties.

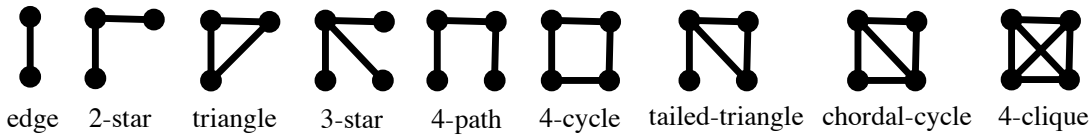
To improve GNNs based on the WL test, k-GNN (Morris et al., 2019) and $3WL-GNN$ (Maron, Ben-Hamu, Serviansky & Lipman, 2019) are proposed respectively based on the k-dimensional Weisfeiler-Leman algorithm(k-WL). They both show that k-WL ($k > 2$) is strictly stronger than 1-WL. Ring-GNN (Z. Chen, Villar, Chen & Bruna, 2019) is another work related to the WL test. Different from above methods, Ring-GNN is based on order-2 G-invariant networks. The authors show that WL test is equivalent to permutation-invariant function approximation. Ring-GNN outperforms order-2 G-invariant networks.

Although k-GNN, 3WL-GNN and Ring-GNN have displayed improved performances, their space and time complexities are considerably high. Therefore, they are often considered less efficient than Spatial GNNs.

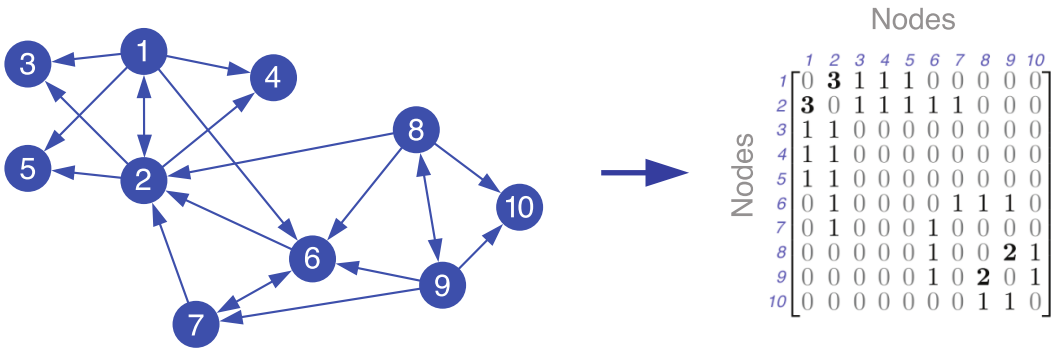
Another body of work aims to exploit the topological information in graphs by



(a) All three-node directed motifs



(b) All undirected motifs of 2-4 nodes



(c) Example of forming a motif adjacency matrix for M_7 in (a)

Figure 2.1: (a) and (c) are adapted from (Benson et al., 2016), (b) is adapted from (J. B. Lee et al., 2019).

considering network motifs. Network motifs are recurring and significant patterns or subgraphs that appear in complex networks more frequently than in randomized networks (Milo et al., 2002). Depending on the type of graphs, motifs can be either directed or undirected. Figure 2(a) and 2(b) shows some examples of directed and undirected motifs respectively. As most motifs involve more than two nodes, they encode informative local structural patterns and also contribute to global properties of the whole graph. A common approach to incorporate network motifs into existing graph learning frameworks is to form motif adjacency matrices (Benson et al., 2016). Given a graph and a certain motif M, we can form the motif adjacency matrix A_M whose entries (i, j) are the number of instances that nodes i and j co-exist in. Figure 2(c) shows an example of the formation of a motif adjacency matrix.

Graph Substructure Network (GSN) (Bouritsas, Frasca, Zafeiriou & Bronstein, 2022) presents a different method to generalise topological information in the model. For node-induced subgraph counting, the number of times a node appears as an endpoint in a subgraph is recorded. For edge-induced subgraph counting, the number of times an edge appears as a boundary in a subgraph is recorded. The embeddings of subgraph counting are passed through the message passing functions as a part of the input features. The authors show that the complexity of GSN is better than k-WL GNNs since the only computationally expensive step is the one-time step of subgraph counting. After that, the complexity of network training and inference is the same as message passing in GNNs. The expressive power of GSN depends on the subgraph used for counting. For example, GSN with 4-clique can distinguish some complex graphs that cannot be distinguished by 3-WL GNNs.

Liu and Song (X. Liu & Song, 2022) show that searching isomorphisms in a graph is equivalent to searching on its dual graph. They then present dual message passing neural networks which learns node features in original graphs and edge features in corresponding dual graphs. The isomorphism counting and matching proceed asynchronously to

reduce computational complexity.

MotifNet (Monti et al., 2018) is dedicated to directed motifs. The authors obtain motif adjacency matrices from directed motifs and then get the motif Laplacian. Although the adjacency matrices are still symmetric, different motifs induce different structures which enables the model to be anisotropic. For the core structure of MotifNet, multivariate polynomials of degree p are applied to the motif Laplacians. Since a large number of coefficients are needed, the polynomials are simplified using two approaches. One is to only consider the two motifs of incoming and outgoing edges. The other is to define the polynomials with recursive products of each degree.

Rossi et al. (R. A. Rossi, Ahmed & Koh, 2018) propose a framework for learning higher-order network embeddings which preserve the interactions between multi-hop neighbouring nodes in the graph. They adopt the same definition of motif adjacency matrices as in (Benson et al., 2016) except that they use undirected motifs. Additionally, the authors propose k-step motif adjacency matrix which is the power of the original motif adjacency matrix. For m motifs with k-step, there are $m \times k$ motif adjacency matrices in total. This embedding method is adopted in Motif Convolutional Networks (J. B. Lee et al., 2019) which uses an attention mechanism and an epsilon greedy strategy to select only one neighbour for each node's information aggregation.

With a main focus on heterogeneous graphs, Motif-CNN (Sankar, Zhang & Chang, 2019) also uses motifs to capture higher-order information. The authors differentiate nodes into different semantic roles according to their types and occurrences in the motifs. Instead of the adjacency matrix in GCN, Motif-CNN uses a motif adjacency matrix weighted by semantic roles to perform convolution operations. Nodes with the same semantic role in all instances of a motif share the same weight. For the weights of different motifs, Motif-CNN adopts the attention mechanism to dynamically adjust the importance of different motifs for each node.

InfoMotif (Sankar, Wang, Krishnan & Sundaram, 2020) also attempts to capture the

features of multi-hop neighbours. The authors propose a concept of attributed structural roles of nodes. Nodes are assigned the same attributed role if they have co-varying attributes in similar motif instances. To identify structurally similar nodes, InfoMotif adopts mutual information maximisation to obtain attribute correlations in motifs.

Reference (H. Li, Wang, Cui, Bai & Hancock, 2020) proposes to incorporate global information with motif-based Subgraph Convolutional Neural Networks. Instead of directly linked neighbours, motif neighbours are used for neighbourhood aggregation. An input graph is transformed to an m-ary tree by cascading each node's m-ary subtree following the order of node degrees. The m-ary tree is then used to generate an adjacency matrix for feature aggregation.

Motif-based Graph Self-supervised learning (Z. Zhang et al., 2021) adopts motifs in predicting molecular properties. The first step involves the molecular graphs being transformed to motif trees where each motif instance is replaced by a node. The motif trees are then used to train the model to learn how motifs are labelled and connected. Specifically, a motif tree is generated by recursively predicting new connections with motifs and the type of new motifs based on existing motif nodes.

The recent work of Motif-based Multiview Graph Attention Networks with Gated Fusion (Piao et al., 2021) and Motif Graph Neural Network (X. Chen et al., 2022) have rather similar structures. They both follow the commonly used message-passing paradigm of GNN framework. Specifically, they first generate motif adjacency matrices and then aggregate node features on each motif adjacency matrix. The key difference between the two pieces of work is how to combine the features aggregated from different motif adjacency matrices. The former uses an attention layer to get a weighted average from the feature matrices before cascading a gated fusion layer to control the weights of residual connections. The latter minimises redundant information among all motifs. Here, redundant information is measured by similarity between node features aggregated from each motif adjacency matrix and those from all other motif adjacency matrices. An

inner product and a non-linear activation function are used for the similarity calculation.

2.5.3 GNNs for Collaborative Filtering

GNNs have become a standard tool for CF by propagating user-item signals over interaction graphs to learn embedding spaces aligned with top-K recommendation. Core designs including message passing, neighbourhood aggregation, and lightweight propagation capitalise on sparse implicit feedback while avoiding heavy feature engineering.

Following the earlier works of GraphSAGE (W. Hamilton et al., 2017) and GCN (T. N. Kipf & Welling, 2017), Neural Graph Collaborative Filtering (NGCF) (X. Wang, He, Wang et al., 2019) is one of the earliest works to apply GCN to CF. The paper proposes a message-passing design for convolution layers with a short circuit mechanism that concatenates the hidden states after each layer as the final output. The datasets used are represented as bipartite graphs so edges only exist between user-item pairs. The message between a user-item pair is the weighted average of the item node embedding and the Hadamard product of the two nodes' embeddings. The convolution layers then aggregate the messages for all nodes to update their embeddings. After obtaining the final node embeddings, the prediction of the interaction between a user-item pair is calculated via the inner product of their embeddings.

Influenced by NGCF, He et al. (X. He et al., 2020) perform ablation studies on NGCF and show that feature transformation matrix and nonlinear activation function do not have additional effectiveness for GCNs on CF tasks. In the same work, an NGCF-inspired model called LightGCN is proposed to eliminate the feature transformation matrix and non-linear activation function in the convolution layers. In a way, LightGCN can be considered a reduced and refined version of the NGCF model while further enhancing experimental performances.

As another key publication, Yu et al. (W. Yu & Qin, 2020) propose Low-pass Collaborative Filter Network (LCFN) that simplifies spectral graph convolution by calculating only a small number of graph Laplacian eigenvectors of small eigenvalues. The work shows that small eigenvalues correspond to low-frequency graph signals which can reflect nodes being connected, while large eigenvalues correspond to high-frequency graph signals which represent noises in data. Through removing high-frequency eigenvectors in graph Laplacian, LCFN avoids high computational cost of eigen-decomposition and eliminates noise in graphs, thereby improves model accuracy and efficiency.

LCFN has a limitation of constructing separate graphs for user and item nodes thus ignores the direct interaction between a user and an item. To address this issue, Low-pass Graph Convolutional Network (LGCN) (W. Yu et al., 2022) uses bipartite graphs to preserve the user-item interactions. Similar to LightGCN, LGCN also simplifies the graph convolution layer by removing feature transformation and non-linear activation. The authors also show that low-pass filter enables LGCN to avoid over-smoothing issue.

Ouyang et al. (Ouyang, Zhang, Hou, Zhang & Ye, 2024) identify that traditional CF models tend to overfit due to noise in labelled training data and propose to mitigate this issue by integrating contrastive learning with a GNN to enhance the alignment and uniformity of user and item representations independently of labels. Alignment is improved by minimizing the distance between representations of interacting user-item pairs. Uniformity is enhanced by distributing these representations more evenly across their respective hyperspheres. Their framework also leverages a novel 0-layer embedding perturbation mechanism for minimal data augmentation, which allows the model to learn these properties in a label-independent manner with faster convergence.

While GNN-based CF improves ranking quality and generalisation, its reliance on simple neighbourhood smoothing limits expressiveness under complex connectivity. These limitations highlight the need for higher-order structure capture and training

objectives that regularise beyond edge-level similarity.

Higher-order GNNs extend beyond immediate neighbours to encode multi-hop information and aim to recover semantics such as shared tastes, cross-category exploration, and co-engagement patterns. Techniques range from multi-layer propagation and polynomial filters to explicit construction of higher-order node connections.

HyperRec (J. Wang, Ding, Hong, Liu & Caverlee, 2020) is a recommendation model that employs hypergraphs to represent higher-order relationships between users and items. The hypergraph is constructed such that each hyperedge represents a group of items that a user has interacted with within a specific time window, which in turn encapsulates the sequential and co-occurrence information in the user's interaction history. By utilising sequential hypergraphs, the model captures complex dependencies between short-term user intent and dynamic item embeddings and offers a richer and more comprehensive user representation compared to traditional user-item graphs.

Ji et al. (Ji et al., 2020) propose a model called Dual Channel Hypergraph Collaborative Filtering which also leverages hypergraphs to capture higher-order relationships among users and items, but form two hypergraphs for user-user and item-item respectively. The hyperedges in a user-user hypergraph is a group of users who share common interactions with one or more items. Similarly, the hyperedges in a item-item hypergraph is a group of items that are connected with at least one common user. Through the two hypergraphs, the model can learn complex higher-order dependencies within users and items.

Zhang and McAuley (H. Zhang & McAuley, 2020) introduce a model that mixes multiple order's higher-order graph convolutions within a stacked structure to capture a broader range of user-item interactions. This hybrid approach is an adaptation of the shortcut technique (K. He, Zhang, Ren & Sun, 2015; K. Xu et al., 2018) as it essentially concatenates interim representations to generate the final embeddings. Similarly, Nguyen et al. (Nguyen, Yu, Nguyen & Yongchareon, 2022) incorporate

higher-order connections within autoencoder framework by augmenting the interaction graph with second-order and fourth-order interactions.

Yu et al. (J. Yu et al., 2021) utilise multi-channel hypergraphs with self-supervised learning strategy. The channels are user views generated from different aspects such as social relationships and common purchases. Auxiliary tasks are then generated from these channels for self-supervised learning. Hypergraph Contrastive Collaborative Filtering (L. Xia et al., 2022) also combine hypergraphs and self-supervised learning for CF. Specifically, hypergraph is constructed based on mutual connections for both users and items. Both the hypergraph and original interaction graph are used to generate contrastive views by edge dropout. The contrastive loss for self-supervised learning maximises the agreement between same nodes in different views and minimises the agreement between different nodes. This objective helps the model refine the embeddings and ensures that similar nodes have closer representations while dissimilar ones are pushed apart.

Xia et al. (J. Xia et al., 2024) introduce a graph signal processing approach that uses two types of filters to separately learn user unique behaviours as well as global behaviours. The user unique behaviours are learned among users that share similar preferences, preventing user-specific preferences from being influenced by the behaviours of dissimilar users. The global behaviours are general interaction patterns across all users, which are captured by a combination of higher-order and ideal low-pass filters. This balances the need to leverage broad user interaction data while avoiding the over-smoothing problem commonly associated with traditional filtering methods.

Gong et al. (Gong, Song, Li & Wang, 2024) design two higher-order neighbour-enhanced strategies to improve GNNs for CF. The embedding propagation (EP)-based strategy and the positive user-item pair (PP)-based strategy. The EP-based strategy employs a novel higher-order graph convolution method that mitigates the excessive decay of higher-order neighbour information by assigning different weight coefficients to

various order neighbours. This design enables the model to capture richer collaborative signals. The PP-based strategy enhances the model's training process by identifying potentially positive samples from a user's higher-order neighbours, based on item similarity and user similarity, to improve the quality of the positive user-item pairs used during model training.

Despite better structural coverage, these higher-order models risk over-smoothing and noise amplification without principled supervision. This motivates incorporating structural semantics into the design of the graph convolution layer.

2.5.4 Graph Contrastive Learning for Collaborative Filtering

Graph Contrastive Learning has emerged as a powerful self-supervised learning paradigm for graph-based recommendation. It focuses on learning robust user and item representations by contrasting multiple views of the interaction graph. This approach addresses data sparsity and improves generalisation without relying on explicit supervision. The foundational work by Wu et al. (J. Wu et al., 2021) introduced SGL, which creates contrastive views through random node and edge dropout. SGL encourages agreement between embeddings derived from these views, which mitigates sparsity and enhancing robustness. However, this stochastic augmentation strategy can distort meaningful graph structures and lead to inconsistencies in semantics and reduced interpretability.

To address these limitations, later models explored simplified or more efficient augmentation strategies. SimGCL (J. Yu et al., 2022) maintains graph connectivity and competitive performance by avoiding explicit structural perturbations and instead injecting Gaussian noise directly into node embeddings. SelfCF (X. Zhou, Sun, Liu, Zhang & Miao, 2023) shifts the focus to augmentation in the output space by applying contrastive learning directly to the final embeddings. By avoiding alteration on the input graph and reliance on negative sampling, this method offers a more lightweight and

efficient training process.

Several models aim to refine the contrastive signal or improve its robustness. BDCL (P. Yu, Bao, Tan & Lu, 2024) applies directional perturbations to preserve sparse behavioural data without random corruption. RecDCL (D. Zhang et al., 2024) introduces both batch-wise and feature-wise contrastive objectives to improve embedding orthogonality. DCCR (R. Zhao, Chen, Sun, Peng & Ju, 2024) uses a self-adaptive noise mechanism and constructs dual collaborative views to extract high-value samples and enhances recommendation diversity. GGDHSCL (X. Liu et al., 2025) introduces a generative diffusion mechanism coupled with hard negative sampling, which strengthens the contrastive signal rather than modifies view structures.

Recent work has also emphasised on improving the handling of noisy or uninformative views. SymmetricGCL (C. Zhao et al., 2025) introduces a denoising strategy using symmetric contrastive pairs to mitigate degradation from noisy augmentations. WeightedGCL (Z. Chen, Xu, Wei & Peng, 2025) enhances SGL with dynamic view-level weighting that allows more informative contrastive views to contribute more significantly to learning.

To explore more stable and informative view construction, NLGCL (J. Xu et al., 2025) generates contrastive views from naturally existing neighbour layers instead of random perturbations, reducing semantic drift while maintaining graph topology. DimCL (C. Zhang et al., 2025) augments representations in a dimension-aware fashion to encourage diverse embedding components for contrastive learning.

To support privacy and personalisation, Personalised Federated Contrastive Learning (S. Wang et al., 2025) adapts GCL to the federated setting by aggregating user-device interactions without centralising user data. This design introduces personalisation and scalability but requires careful calibration to maintain embedding consistency across clients.

Together, these works illustrate a trajectory from random augmentations toward

more principled and robust contrastive learning techniques. Yet, many still operate with implicit structure and offer limited topological semantics, which motivates structurally grounded alternatives.

Recognising the limitations of uniform random perturbations, several researchers have explored structured or semantically informed augmentations. SG-CLR (Y. Zhang et al., 2024) enhances the semantic integrity of contrastive views by guiding edge additions based on homophily. HGCL (J. Xue et al., 2025) takes a hierarchical perspective by clustering items into multiple resolution layers to preserve coarse-to-fine structural semantics. NCL (Z. Lin et al., 2022) advances the idea of local structure by treating topological neighbours as natural positives rather than sampling corrupted ones. This strategy captures contextual consistency and reduces semantic drift during augmentation.

JCG (Dai, Li, Nong, Bi & Chu, 2024) combines structural and semantic contrastive objectives to address data imbalance and reduce the influence of high-degree nodes. This improves generalisation across heterogeneous graph distributions. MD-GCCF (X. Li, Tian, Dong & Ji, 2024) advances multi-view learning by integrating deep collaborative signals and designing both local and global contrastive objectives. By mitigating over-smoothing and reducing redundancy, this architecture yields more informative representations for recommendation.

Several models introduce learnable or deterministic transformation schemes. CL-KDM (Ni, Xiong, Zheng & Wang, 2024) bypasses traditional dropout strategies entirely by maximising statistical dependence between views, which preserves global latent geometries without edge removal. LMACL (X. Liu et al., 2024) combines GCN and GAT to learn augmentation strategies to generate multi-hop and neighbour-aware contrastive views.

These methods present promising steps toward augmentations that encode high-quality semantics or structure. However, many rely on heuristics or predefined rules and

rarely integrate clear higher-order graph patterns such as motifs into the view generation process.

2.5.5 Motif-Based GNNs and Methods for Collaborative Filtering

A series of recent works show that the topological structure of graphs has significant influence on local and global representation learning (K. Xu et al., 2019; Bouritsas et al., 2022). As one way of exploiting the topological information in graphs, network motifs have been used in a range of research works for GNNs. In this section, a range of motif-based GNNs will be explained.

As one of the first works using motifs in GNNs, MotifNet (Monti et al., 2018) uses motifs for directed graphs. The work obtains motif adjacency matrices from directed motifs and then get the motif Laplacian. Although the adjacency matrices are still symmetric, different motifs induce different structures which enables the model to be anisotropic. For the core structure of MotifNet, multivariate polynomials of degree p are applied to the motif Laplacians. Due to the large number of coefficients required, the polynomials are simplified using two approaches. One is to only consider the two motifs of incoming and outgoing edges. The other is to define the polynomials with recursive products of each degree.

Rossi et al. (R. A. Rossi et al., 2018) propose a framework that uses motifs to learn higher-order network embeddings which represent interactions between multi-hop neighbours. They adopt the same definition of motif adjacency matrices as in (Benson et al., 2016) except that they use undirected motifs. Additionally, the authors propose k-step motif adjacency matrix which is the power of the original motif adjacency matrix. For m motifs with k steps, there are $m \times k$ motif adjacency matrices in total. This embedding method is adopted in Motif Convolutional Networks (J. B. Lee et al., 2019) which uses an attention mechanism and an epsilon greedy strategy to select only one

neighbour for each node's neighbourhood aggregation.

With a focus on heterogeneous graphs, Motif-CNN (Sankar et al., 2019) also uses motifs to capture higher-order information. The authors differentiate nodes into different semantic roles according to their types and occurrences in the motifs. Instead of the adjacency matrix in GCN, Motif-CNN uses a motif adjacency matrix weighted by semantic roles to perform convolution operations. Nodes with the same semantic role in all instances of a motif share the same weight. For the weights of different motifs, Motif-CNN adopts the attention mechanism to dynamically adjust the importance of different motifs for each node. Similarly, InfoMotif (Sankar et al., 2020) also captures multi-hop interactions using structural roles of nodes in motifs. Nodes are considered as the same attributed role if they have co-varying attributes in similar motif instances. To identify structurally similar nodes, InfoMotif adopts mutual information maximisation to obtain attribute correlations in motifs.

Li et al. (H. Li et al., 2020) propose to incorporate global information with motif-based Subgraph Convolutional Neural Networks. Instead of directly linked neighbours, motif neighbours are used for neighbourhood aggregation. An input graph is transformed to an m-ary tree by cascading each node's m-ary subtree following the order of node degrees. The m-ary tree is then used to generate an adjacency matrix for feature aggregation.

The recent works of Motif-based Multiview Graph Attention Networks with Gated Fusion (Piao et al., 2021) and Motif Graph Neural Network (X. Chen et al., 2022) have rather similar structures. They both follow the commonly used message-passing paradigm of GNN framework. Specifically, they first generate motif adjacency matrices and then aggregate node features on each motif adjacency matrix. The key difference between the two pieces of work is how the features aggregated from different motif adjacency matrices are combined. The former uses an attention layer to get a weighted average from the feature matrices before cascading a gated fusion layer to control the

weights of residual connections. The latter minimises redundant information among all motifs. Here, redundant information is measured by the similarity between node features aggregated from each motif adjacency matrix and those from all other motif adjacency matrices. An inner product and a non-linear activation function are used in the similarity calculation.

Motifs are also used in graph classification. Zhang et al. (S. Zhang, Hu, Subramonian & Sun, 2024) present a novel framework that uses motifs in GNNs pre-training for molecular property prediction. The framework learns subgraph embeddings by maximising the likelihood of each subgraph belonging to a motif, and then optimises the embeddings by combining them to contrast with initial who-graph embeddings. Similarly, Zhou et al. (Z. Zhou, Zhou et al., 2024) also use motifs to learn subgraph embeddings for graph classification. Differently, they rank subgraph importance by calculating the similarity between subgraph embeddings and a graph-level vector, and combine important subgraphs into the overall graph embedding. In the subgraph-motif alignment phase, top candidate subgraphs are selected based on importance scores, and motifs are iteratively extracted using k-means clustering. These motifs guide the alignment of subgraph representations through a contrastive loss function and ensures that subgraphs align with motifs of their class while remaining distinct from others.

Despite the rich collection of motif applications in GNNs, there is currently very limited research on motif-based CF. This gap motivates Chapter 3, which integrates a motif attention mechanism that selectively aggregates higher-order, motif-defined neighbours. It also motivates Chapter 4, which embeds motifs directly into graph convolution to achieve efficient single-layer higher-order aggregation with improved adjacency construction for all bipartite motifs up to four nodes. Together, these chapters provide CF-specific motif building blocks that address missing structure, targeted aggregation, and computational practicality.

Recent studies have explored motif-based enhancements in recommender systems.

Motifs-Res (Y. Sun, Zhu, Du & Tian, 2022) formulates motif co-occurrence as hyperedges and applies hypergraph convolution with contrastive loss to exploit local motif semantics. MotifSR (Cui, Cai, Wu, Ma & Wang, 2021) incorporates motif-enhanced temporal subgraphs to better capture session-level user behaviours in sequential recommendation. These models demonstrate the growing interest in exploiting motif semantics to strengthen signal quality and model generalisation in graph-based recommendation.

Complementing this, motif-based contrastive learning has been studied in other domains. Motif-driven contrastive learning of graph representations (S. Zhang et al., 2024) introduces motif-induced views to enforce global and local consistency in general graph representation learning, where motif co-occurrence defines semantically consistent neighbourhoods. MotifCC (X. Wu, Wang, Lin, Xi & Yu, 2024) proposes a framework that builds subgraphs from triangle motifs and applies contrastive learning between lower-order and higher-order views to jointly preserve structural uniqueness and encourage clustering separability. MHGCL (C. Li et al., 2025) refines the selection of contrastive pairs by jointly leveraging motif structures and node homogeneity, which suppresses noise and improves representation discrimination in general graph tasks. Although effective, these works do not target recommender systems and typically operate in node classification or community detection settings.

Our work in Chapter 5 departs from the above by being the first to integrate motif-based structures into contrastive view generation specifically for collaborative filtering. Instead of treating motifs as auxiliary enhancements or post-processing filters, we construct motif-induced adjacency matrices and use them as structurally meaningful and interpretable views. Rectangle motifs capture higher-order co-engagement signals, while the combination of star motifs additionally includes sparse preferences or transient behaviour. By contrasting these motif-guided views, our approach prioritises robust local structures, reduces potential noise arisen from sparse or isolated patterns, and

facilitates transparent recommendation. This approach uniquely unifies interpretability, semantic augmentation, and robustness within the contrastive learning framework for graph-based recommendation.

2.5.6 Heterophily-Aware GNNs and Methods for Collaborative Filtering

Most GNNs follow a homophily assumption in their architecture design (X. Zheng et al., 2022) where similar nodes tend to connect each other while dissimilar nodes are less likely to be adjacent. In contrast, heterophily in graphs means dissimilar nodes are more likely to be connected. When working on some real world datasets that exhibit heterophily rather than homophily, many of these models face the challenge of performance degradation. To address this, a series of research efforts have been made on specifically designed GNNs for heterophilic graphs.

To address the challenge of learning under low-homophily, H2GCN (J. Zhu et al., 2020) emphasises ego-neighbour embedding separation, higher-order neighbourhood aggregation, and the combination of intermediate representations. The model identifies these as essential design principles that enable GNNs to generalize beyond homophilic structures. In paper (J. Zhu et al., 2021), the authors propose CPGNN, which integrates an interpretable compatibility matrix to represent the homophily or heterophily level between connected nodes. This matrix enables the model to generalise across both homophilic and heterophilic graphs. To improve adaptability in spectral space, FAGCN (Bo, Wang, Shi & Shen, 2021) introduces a self-gating mechanism that adaptively fuses homophilic and heterophilic signals, demonstrating that relying solely on homophilic information limits representational expressiveness. In a similar direction, Chanpuriya and Musco (Chanpuriya & Musco, 2022) simplify the GCN architecture and extend it to heterophilic settings, showing that heterophily-aware adjustments can yield strong

performance with minimal computational cost.

Li et al. (X. Li et al., 2022) present GloGNN and GloGNN++, which generate node embeddings through global correlation-based neighbourhood aggregation rather than local proximity. By learning a signed coefficient matrix in closed form, their models recover global homophily patterns even when local edges are misleading. To address multi-hop noise, Yang et al. (T. Yang et al., 2022) design Graph Pointer Neural Networks (GPNN), which employ a pointer mechanism to select and order informative multi-hop neighbours, mitigating over-smoothing and irrelevant information propagation. Zhou et al. (S. Zhou, Guo, Aggarwal, Zhang & Wang, 2022) propose DisenLink for link prediction on heterophilic graphs and introduces disentangled representation learning to capture multiple latent factors that drive edge formation. Each node embedding is decomposed into factor-specific components to enable factor-aware message passing and better link prediction.

In unsupervised graph representation learning, Liu et al. (Y. Liu, Zheng, Zhang, Lee & Pan, 2023) present GREET, which discriminates between homophilic and heterophilic edges via an edge-level discriminator and learns through dual-channel contrastive objectives. This approach enhances robustness across heterophilic networks and improves transferability without supervision.

Yan et al. (Yan, Hashemi, Swersky, Yang & Koutra, 2022) unify heterophily and over-smoothing as two aspects of the same underlying problem. They define node heterophily metrics to explain performance degradation in GCNs and then propose GGCN, which applies structure-based and feature-based edge correction strategies to mitigate both issues. From a theoretical perspective, Wang et al. (J. Wang, Guo, Yang & Wang, 2024) formalise a Heterophilous Stochastic Block Model (HSBM) and show that the separability gain of a GCN-style operator is governed by the ℓ_2 distance between neighbourhood label distributions and by $\sqrt{\mathbb{E}[\deg]}$, yielding a trichotomy of good/mixed/bad heterophily, quantifying the impact of neighbourhood inconsistency,

and explaining how depth can preserve relative separability under over-smoothing.

From a topological perspective, Bodnar et al. (Bodnar, Di Giovanni, Chamberlain, Lió & Bronstein, 2022) propose Neural Sheaf Diffusion, a framework that models the consistency of features across edges using sheaf theory. This formulation improves expressiveness on heterophilic graphs and alleviates over-smoothing by enforcing localized but coherent diffusion. Luan et al. (Luan et al., 2022) revisit conventional homophily metrics and introduce Adaptive Channel Mixing (ACM), which dynamically combines aggregation, diversification, and identity channels to adapt to node-wise heterophily. By learning how to mix these channels, ACM captures richer local patterns across diverse graphs. Similarly, Rossi et al. (E. Rossi et al., 2023) show that incorporating edge directionality increases effective homophily and propose Dir-GNN, a general framework that separately aggregates incoming and outgoing messages, achieving large improvements on heterophilic benchmarks while maintaining stability on homophilic ones. Wang, Solin, and Garg (H. Wang, Solin & Garg, 2025) develop Heterophily-Informed Message Passing (HetMP), which learns a similarity signal to modulate messages across homophilous, heterophilous, and original channels with optional mixing which preserves both low- and high-frequency components and further extending to a flow-based generative model called HetFlows.

To automate heterophily-aware design, Zheng et al. (X. Zheng et al., 2023) develop Auto-HeG, the first neural architecture search (NAS) framework specialized for heterophilic graphs. Auto-HeG integrates heterophily into its search space, supernet training, and architecture selection stages, yielding specialized architectures that outperform manually designed models. Qiu et al. (C. Qiu et al., 2024) introduce Latent Homophilic Structures (LHS), which iteratively refine graph topology via a self-expressive multi-node interaction model and a dual-view contrastive learning mechanism. The refined structure enables GCNs to aggregate information in a homophilic manner even over heterophilic graphs. Zheng et al. (Y. Zheng, Xu & Chen, 2024) further propose

HiGNN that leverages heterophilic information itself by constructing an auxiliary graph that connects nodes sharing similar neighbor distributions, which improves semantic connectivity and representation learning.

Fabbri et al. (Fabbri, Croci, Bonchi & Castillo, 2022) introduce an agent-recommender co-evolution simulator that varies group size, homophily level, and link-recommendation strategies to study long-term exposure dynamics. The study shows that when a minority is homophilic it gains disproportionate visibility, whereas a heterophilic minority becomes underexposed. Moreover, link recommenders strengthen individual-level exposure inequality independent of group homophily. This evidence motivates heterophily-aware modelling in recommendation, where algorithms explicitly account for how mixing patterns interact with learning and feedback loops.

Peng et al. (S. Peng, Sugiyama & Mine, 2022) propose a spectral reweighting framework that emphasizes informative graph-frequency components for collaborative filtering. By preserving high-frequency signals that typically encode cross-community or dissimilar-neighbour information, the method mitigates over-smoothing and retains predictive cues characteristic of heterophilic neighbourhood. In effect, frequency-selective propagation operates as a practical surrogate for heterophily-awareness when dissimilar relations carry complementary information.

Jiang et al. (W. Jiang, Gao, Xu, Chen & Yin, 2024) introduce SHaRe, a data-centric social recommendation framework that diagnoses preference-aware homophily and rewires the social graph by adding highly aligned ties while pruning heterophilic ones. To address representation drift during rewiring, SHaRe integrates a contrastive calibration objective. It improves accuracy across homophily ratios and enhances a range of graph-based recommenders. This line of work shares a similar strategy of heterophily-awareness that models measure local alignment and then modify structure and training signals accordingly.

In (Gholinejad & Chehreghani, 2024), the authors present HetroFair, a fairness-oriented GNN recommender that combines fairness-aware attention with heterophily feature weighting to assign distinct aggregation weights under heterophilic conditions. Experiments across multiple datasets indicate simultaneous gains in item-side fairness and user-side accuracy. This shows the abilities of retaining informative dissimilar neighbours and mitigating popularity-driven exposure disparities.

Despite these advances, current approaches either rewire graphs toward latent homophily (W. Jiang et al., 2024), rely on frequency selection that implicitly preserves heterophilous cues (S. Peng et al., 2022), or couple propagation with fairness controls (Gholinejad & Chehreghani, 2024) while offering limited mechanisms to explicitly supervise and exploit heterophily at fine-grained structural units. In particular, existing social-exposure analyses (Fabbri et al., 2022) highlight the stakes but do not provide a motif-level training signal that distinguishes homophilic, mixed, and heterophilic interactions during contrastive learning and propagation. To address this gap, we propose MoHeGCL in Chapter 6, which treats user-anchored triads as contrast units with soft motif-type labels, and employs a motif-conditioned training strategy to leverage heterophilic structure for recommendation.

2.6 Summary

This chapter has examined four areas of research that support our research in this thesis: foundational recommender system methods, mashup-API recommendation, network motifs, and graph neural networks. Classical approaches such as content-based filtering and collaborative filtering established the core principles of modelling user-item interactions, but also revealed persistent limitations including sparsity, cold start, and limited capacity to capture complex structural signals. Research on mashup-API recommendation further highlighted the challenges of modelling on bipartite graphs,

where API composition, co-invocation behaviour, and domain semantics play critical roles.

The review of network motifs demonstrated the importance of higher-order structures in revealing recurrent meaningful patterns across diverse networks. Motifs offer a principled approach to encode local connectivity beyond pairwise relations, and their incorporation into graph representation learning has shown promise for improving expressiveness.

Graph neural networks and their extensions have significantly advanced the ability to learn representations from graph-structured recommendation data. However, existing models often assume homophily, ignore higher-order structure or lack robustness in heterophilic scenarios.

Taken together, the literature reveals a clear gap of needing recommendation models that capture higher-order graph local structural information and utilise it to obtain performance improvements, interpretability, or heterophily awareness. These insights directly inform the design of the proposed models presented in the following chapters.

Chapter 3

Motif-Based Graph Attention Network for Recommendation

3.1 Introduction

The development of Web services allows information spread faster and more diversely than ever. As a result, it has become more challenging to design a Web service recommender system that can fast and accurately select ideal Web APIs from the vastly expanded resources. To improve the performance of recommender systems, researchers have adopted many deep learning methods that show great performance in other research areas.

Recently, graph neural networks (GNNs) have emerged as a type of ideal solution to graph learning tasks. Compared to deep neural networks that only process direct interactions between mashups and APIs(T. Liang, Chen, Wu, Dong & Bouguettaya, 2016; Xie, Wang et al., 2019; Xie, Chen, Lin, Zheng & Lin, 2019) or only use co-invocations for model initialisation(Ma, Geng & Wang, 2021; Yao et al., 2021), GNNs can continuously capture the higher-order information through multiple propagation layers to generate accurate embeddings for collaborative filtering.

A series of recent advancements in recommender systems(X. Wang, He, Wang et al., 2019; X. He et al., 2020; W. Yu et al., 2022) that adopt graph convolutional network (GCN(T. N. Kipf & Welling, 2017)) show great potential of this approach. A widely acknowledged issue of GCN is that after several layers most nodes' embeddings become dominated by a small number of high-degree nodes(K. Xu et al., 2018), so called over-smoothing. A solution to this issue is to combine additional structural information such as network motifs into layer propagation to enhance the degree of node interaction within each layer so that the number of layers can be reduced.

Network motifs are recurring and significant patterns or subgraphs that appear in complex networks more frequently than in randomised networks (Milo et al., 2002). By incorporating motifs in layer-wise propagation, nodes become aware of the local structure and the weights of neighbours are redistributed according to their appearances in different motifs. So far, this technique is limited to graph learning tasks on unipartite graphs and to eliminate the redundancy in multiple motifs researchers have tried enhance the methods with techniques such as attention mechanism(Monti et al., 2018; Piao et al., 2021) and mutual information maximisation(X. Chen et al., 2022). For recommender systems, the data are mostly represented by bipartite graphs so the types of motifs are much less than in unipartite graphs, which means motif-based GNNs built for bipartite graphs can be more efficient.

In this chapter, we first explore the common motifs in the user-item bipartite graph and identify all the seven up-to-four-node motifs in bipartite graphs. Based on the generated motif adjacency matrices, we propose a motif-based graph attention mechanism that aggregates higher-order information in the motifs. We then propose a motif-based graph attention network for service recommendation (MGSR).

The main contributions of this work are as follows.

- We identify and define the various connectivity structures as motifs in bipartite

graphs.

- We propose a Motif-based Graph Attention Network for Service Recommendation that aggregates higher-order information of motif-based neighbours into the embedding progress for graph propagation.
- We conduct experiments on the ProgrammableWeb dataset and the results demonstrate that MGSR outperforms state-of-the-art baselines.

3.2 Related Work

3.2.1 Mashup-API Recommendation

Mashup-API composition recommendation aims to suggest sets of APIs that are frequently used together to help developer design applications efficiently. Most publications use data from ProgrammableWeb (*ProgrammableWeb has been retired*, n.d.) or similar API portals. The literature naturally divides into traditional and graph-based methods. Traditional methods include content-based filtering, matrix factorisation, topic models, and random-walk retrieval while graph-based methods include graph auto encoders and GNNs.

Traditional Methods

Early approaches for mashup-API recommendation rely primarily on textual and descriptive metadata provided by platforms such as ProgrammableWeb. APIs and mashups are represented using textual descriptions, tags, functional categories, and provider information. Recommendations are generated by computing similarity between the functional intent of a target mashup and candidate APIs. Topic modeling methods such as Latent Dirichlet Allocation (LDA) were widely used to uncover latent functional themes and to match these themes to mashup requirements (Z. Gao et al., 2016).

More recently, semantic representation models have been introduced to improve the expressiveness of these text embeddings. For example, Y. Wang, Chen, Huang, Xia and Jiang (2023) proposes a framework that leverages pre-trained transformer encoders to derive contextualised semantic representations of both mashup requirements and API descriptions. By learning these richer embeddings, the model captures functional intent more accurately than traditional TF–IDF or LDA approaches. It then applies neural matching to score API candidates, demonstrating substantial gains in recommendation accuracy within the ProgrammableWeb ecosystem.

Another traditional line of work exploits co-usage information derived directly from mashup construction records (W. Gao et al., 2015). If two APIs frequently appear together across many mashups, then they are assumed to be complementary and therefore good candidates for joint recommendation. The underlying assumption is that mashup developers reveal implicit composition knowledge through their usage behaviour. However, recent analysis of correlation-graph-based methods (H. Li et al., 2017) shows that such co-occurrence signals are strongly affected by popularity bias, leading popular APIs to dominate recommendations even when they are not the most suitable choices. As a result, later methods began to consider weighted or context-sensitive co-usage signals to mitigate these effects although these strategies still remained within the broader paradigm of frequency-driven recommendation.

A third stream of research formalises mashup recommendation as a matrix completion task using the mashup–API invocation matrix as the predictive target. Here, collaborative filtering and matrix factorisation techniques are used to infer missing associations and to predict which APIs are likely to be suitable for a mashup currently under development. This formulation allows the system to learn latent representations of mashups and APIs that capture functional interaction patterns beyond direct co-usage. W. Xu, Cao, Hu, Wang and Li (2013) builds coupled matrices for users, mashups and services and use a coupled matrix factorisation model to make predictions from the

invocation matrix and social context information. Yao, Wang, Sheng, Ruan and Zhang (2015) proposes a probabilistic matrix factorisation method over the mashup–service composition matrix with an additional regulariser that captures implicit service correlations. M. M. Rahman, Liu and Cao (2017) explicitly models the mashup-API invocation matrix and applies matrix factorisation to predict unseen mashup–API invocations. These matrix factorisation models established a crucial foundation for later neural models by demonstrating that mashup recommendation requires combining observed usage evidence with latent relational structure.

Graph-Based Methods

Research on mashup-oriented API recommendation has evolved through several methodological phases, which reflects increasingly sophisticated assumptions about how service ecosystems encode functional compatibility. Early methods treated the problem mainly as a similarity-based retrieval task. Later extensions include relational modeling of heterogeneous service networks, structural reasoning over higher-order graph patterns and combining semantic and structural signals.

Traditional similarity-based methods use tag similarity to match mashup requirements with APIs but are proved to be insufficient for capturing the complex relationships between services. Recognising that mashups, APIs and tags form a multi-class network, researchers reframed the problem as heterogeneous information networks (HINs). In one of the early influential studies, T. Liang et al. (2016) shows that representing APIs and related entities as a typed network and computing similarity along structured semantic paths such as API-Tag-API significantly improved functional matching accuracy. Meta-paths enabled the system to embed domain knowledge such as categories and developer usage patterns directly into the similarity computation, which allows the model learn better node representations.

While meta-paths improve structured similarity, they have to be pre-defined and

remained limited on sparse data. To address the limitations, later research applies representation learning on heterogeneous networks. Lima and Liu (2025) demonstrates that network embedding methods could preserve meta-path-based proximity in a latent space. This enables the model to generalise relational patterns even on sparse data. By treating meta-paths as supervising signals during training phase, this approach captures latent compatibility relationships that are not directly observable from local interactions, therefore improving coverage for long-tail APIs.

Building on top of HIN approaches, recent work recognised that service networks exhibit global structural behaviour that are not fully utilised when training models only on local mashup-API invocatioln graphs. To better exploit these patterns such as recurring compositions and domain-specific API communities, M. Tang et al. (2024) introduces a pre-training and fine-tuning framework that pre-trains node embeddings on the full HIN graph before making recommendations. The model captures long-range relational dependencies and improves robustness for cold-start APIs.

Relational HIN modeling captures typed connections but not the intensity of co-usage relationships. A separate research direction thus models mashups as API co-usage graphs and applies techniques inspired by Neural Graph Collaborative Filtering (X. Wang, He, Wang et al., 2019). Lian and Tang (2022) proposes to propagate co-usage signals through a mashup-API bipartite graph, which enables the model to capture higher-order proximity where APIs become compatible through shared or multi-hop usage contexts. Unlike simple co-occurrences, these graph-based methods learn deeper structural dependencies among APIs and offers superior performance especially when direct co-usage data are limited.

Another group of methods model mashups, APIs, and their tags as a knowledge graph, which is then used for retrieval-based recommendation via random walks. Random walk with restart is commonly used to propagate a relevance signal from a mashup query to candidate APIs through graph connectivity (Yoon, Jin & Kang, 2018). X. Wang,

Wu and Hsu (2019) builds a mashup–API knowledge graph and perform random walk with restart to rank relevance scores of candidate APIs. X. Wang et al. (2021) runs deep random walks on a mashup–API knowledge graph to learn embeddings and then estimates relevance between mashup requirements and APIs via these graph-propagated representations. However, these approaches rely heavily on explicit graph structure and do not learn latent representations, which limits their flexibility when the ecosystem evolves or when connectivity is sparse.

A known challenge in co-usage graphs is that popular APIs dominate neighbourhoods and distorts compatibility signals. To address this, attention-based models selectively weight neighbours according to their contextual relevance. Shen, Wang, Zhang and Chen (2023) introduce a graph attention mechanism that distinguishes between meaningful co-invocation relationships and incidental edges created by high-degree APIs. By learning to prioritise informative neighbours, the model shifts from naive proximity modelling to compatibility-oriented neighbourhood aggregation, which reduces noise and improves the recommendation quality.

The most recent generation of methods integrates textual semantics with graph structural reasoning in unified end-to-end neural architectures. API descriptions contain rich functional information but are often inconsistent, sparse, or noisy. Graph structure encodes compatibility but cannot capture meaning alone. T. Yu et al. (2023) enhances graph neural networks with BERT-based encoding to jointly learn semantic relevance, relational structure, and compatibility patterns. This approach resolves limitations of text-only or graph-only methods and demonstrate better robustness under real-world settings where both semantic and structural information are essential.

3.2.2 Motif-Based Graph Neural Networks

A growing body of work demonstrates that graph topology plays a crucial role in shaping both local and global node representations (K. Xu et al., 2019; Bouritsas et al., 2022). Network motifs are small recurring subgraph patterns appearing in real-world data more frequently than in random networks. Motifs provides a solid theoretical guidance for capturing higher-order structure and have therefore become an important building block in a range of GNN architectures. This section reviews motif-based GNN methods and is organised according to their conceptual development.

The use of motifs in GNNs begins with MotifNet (Monti et al., 2018), one of the earliest models to directly incorporate motif structures into graph convolution. MotifNet constructs motif-specific adjacency matrices from directed motifs and derives corresponding motif Laplacians. This method enables anisotropic filtering even though the matrices remain symmetric. To perform graph convolution, multivariate polynomial filters of degree p are applied to these motif Laplacians. Considering these filters introduce a large number of parameters, the authors propose two simplifications. On one hand the model is restricted to incoming and outgoing motifs. On the other hand, the polynomial terms are calculated recursively. This work established motifs as a structural basis for directional and context-aware graph filters.

R. A. Rossi et al. (2018) introduce a more general framework for learning higher-order node embeddings from undirected motifs. Following the motif adjacency formulation proposed by Benson et al. (2016), they further propose k-step motif adjacency matrices obtained by raising each motif adjacency operator to the kth power. With m motifs and k propagation steps, the framework yields $m \times k$ structural channels that capture multi-hop relationships. These motif-derived embeddings are subsequently utilised in Motif Convolutional Networks (J. B. Lee et al., 2019), where an attention mechanism together with an ϵ-greedy neighbourhood selection strategy determines which of the

many motif-based structural signals should be propagated during aggregation.

Motif learning has also been extended to heterogeneous graphs. Motif-CNN (Sankar et al., 2019) adapts motif-based convolution to heterogeneous networks by assigning nodes to semantic roles based on their positions within motif instances. Instead of using the standard adjacency matrix, Motif-CNN constructs role-weighted motif adjacency matrices and applies convolution over these role-aware structures. Motifs are further combined using an attention mechanism that selects the most relevant motif type for each node. A related approach, InfoMotif (Sankar et al., 2020), focuses on discovering meaningful structural roles by identifying sets of attributes that co-vary across motif instances. Through mutual-information maximisation, InfoMotif learns role-aware representations that capture multi-hop and attribute-based interactions beyond simple structural patterns.

To integrate motif information with a more global notion of graph structure, H. Li et al. (2020) propose LGL-GNN, which introduces motif-based subgraph convolution through an m-ary tree transformation of the input graph. Rather than aggregating information solely from adjacent nodes, LGL-GNN uses motif-induced neighbours to construct m-ary subtrees ordered by node degree. These subtrees are subsequently converted into adjacency matrices that enable the model to incorporate broader structural context during feature aggregation.

More recent research continues to expand the use of motifs by treating each motif type as a distinct structural view. Both the Motif-based Multiview Graph Attention Network with Gated Fusion (Piao et al., 2021) and the Motif Graph Neural Network (X. Chen et al., 2022) adopt the message-passing paradigm, which generates separate feature representations from each motif adjacency matrix. Their main contribution lies in the fusion mechanism. The former employs an attention layer followed by gated fusion to regulate how motif-specific features and residual connections are combined. In contrast, the latter reduces redundancy among motif channels by minimising similarity

across their aggregated features and used an inner-product similarity measure with non-linear activation. These models demonstrate the effectiveness of multi-view motif reasoning when combined with sophisticated channel integration schemes.

3.3 Background

A mashup-API network can be represented by a bipartite graph $G = (U, V, E)$ where U denotes the set of mashups, V denotes the set of APIs, and $E = U \times V$ with $e_{ij} \in E$ is an edge between U and V. If an API i is invoked by a mashup j then $e_{ij} = 1$ and zero otherwise.

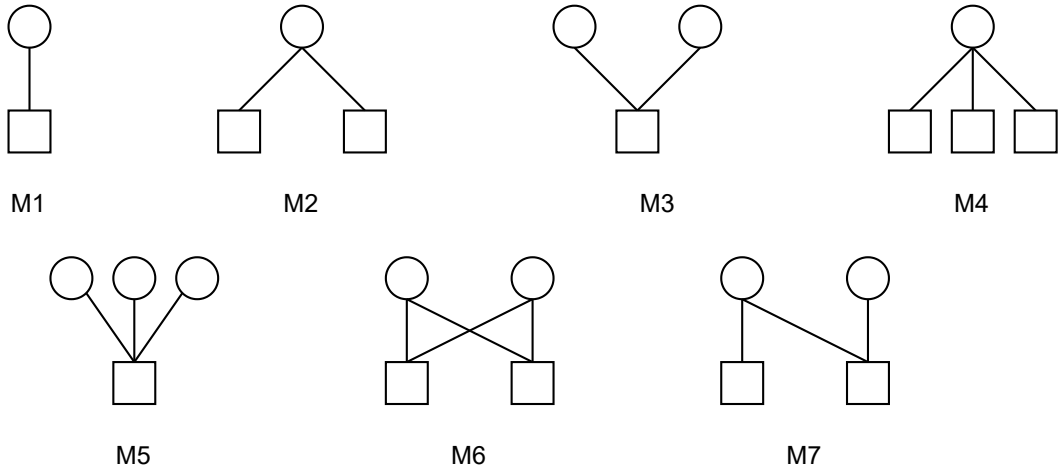


Figure 3.1: Motifs of two to four nodes in bipartite graphs

Due to the absence of mashup-mashup edges and API-API edges, the types of motifs in bipartite graphs are significantly less than in unipartite graphs. For the balance between the importance of local structure and higher-order neighbourhood connection as well as the limitation of computational resources, we decide to limit the maximal size of motifs within four nodes, by which we have seven motifs in total. Figure 3.1 shows the seven motifs $M_1, M_2, ..., M_7$ used in this chapter.

As contributed in the paper (G. Wang et al., 2023), a proposition and its proof are

shown below.

Proposition 1. *Let $G = (U, I, E)$ be a bipartite graph, then*

$\{M_1, M_2, M_3, M_4, M_5, M_6, M_7\}$ is a complete set of motifs consisting four nodes at most in G.

Sketch of Proof. Let n_u, n_i be the number of nodes from U and I, then the maximum number of possible edges can be calculated as $e_{max} = n_u \times n_i$ when these nodes form a complete bipartite graph. The minimum number of edges for a connected bipartite graph can be calculated as $e_{min} = n_u + n_i - 1$. To form a motif with these nodes, one can choose any n_e edges from the e_max edges provided $n_e \geq e_{min}$. Here we provide the exhaustive cases for motifs consisting of four nodes at most:

Case 1: If $n_u = 1, n_i = 1$, then $e_{max} = e_{min} = 1$. The number of possible motifs are $\binom{1}{1} = 1$. We refer to this motif as M_1.

Case 2: If $n_u = 1, n_i = 2$, then $e_{max} = e_{min} = 2$. The number of possible motifs are $\binom{2}{2} = 1$. We refer to this motif as M_2.

Case 3: If $n_u = 2, n_i = 1$, then $e_{max} = e_{min} = 2$. The number of possible motifs are $\binom{2}{2} = 1$. We refer to this motif as M_3.

Case 4: If $n_u = 1, n_i = 3$, then $e_{max} = e_{min} = 3$. The number of possible motifs are $\binom{3}{3} = 1$. We refer to this motif as M_4.

Case 5: If $n_u = 3, n_i = 1$, then $e_{max} = e_{min} = 3$. The number of possible motifs are $\binom{3}{3} = 1$. We refer to this motif as M_5.

Case 6: If $n_u = 2, n_i = 2$, then $e_{max} = 4, e_{min} = 3$. If $n_e = 4$, the number of possible motifs are $\binom{4}{4} = 1$. We refer to this motif as M_6. If $n_e = 3$, the number of possible motifs are $\binom{4}{3} = 4$. From Figure 3.1 we can see that the four possible motifs are isomorphic. Hence we only have one motif, denoted as M_7, for $n_e = 3$.

□

To exploit the structural information in motifs, we generate motif adjacency matrices

following (Benson et al., 2016) where a motif adjacency matrix's entry (i, j) are the number of motif instances that nodes i and j co-exist in. Figure 2 shows an example of a motif and its corresponding motif adjacency matrix generated from the graph. With motif adjacency matrix, we can then define motif neighbours as nodes that appear in at least one instance of the motif in the graph.

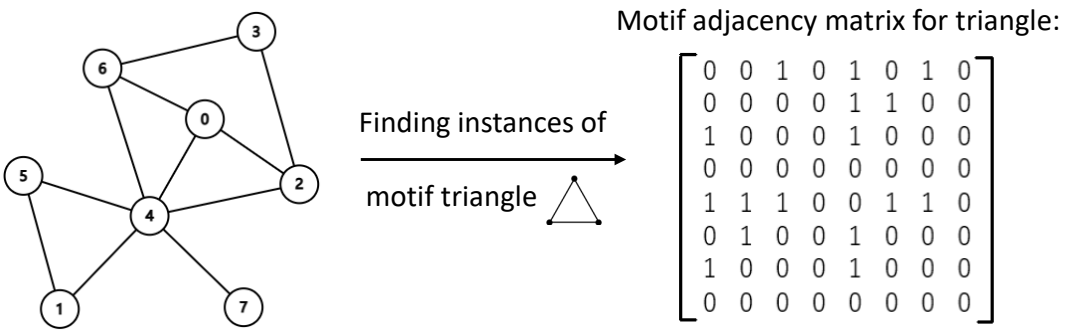


Figure 3.2: Example of forming a motif adjacency matrix.

The proposed motif attention mechanism is based on the graph attention mechanism used in graph attention networks (GAT)(Veličković et al., 2018). Graph attention is the importance of a node to its neighbours. It essentially measures the similarity between each pair of adjacent nodes. It is calculated between each node and its neighbours in convolution layers before being used for weighted neighbourhood aggregation. The node-level calculation of attention weight in a single convolution layer is given by the following equations:

$$\alpha_{ij} = \mathrm{softmax_i}\left(\mathrm{LeakyReLU}\left(\vec{a}^T[Wh_i \| Wh_j]\right)\right)$$

where α_{ij} is the attention for node i from node j. Node i, j's embeddings $h_i, h_j \in \mathbb{R}^{n\times d}$ are first transformed with a linear transformation matrix $W \in \mathbb{R}^{d\times d}$ and are then concatenated and multiplied by a vector $\vec{a} \in \mathbb{R}^{2d}$ to generate a similarity score between node i and j, where n is the number of nodes and d is the dimension of node embeddings.

The score is then fed to a LeakyReLU function and a softmax function to produce a percentage that measures the importance of node j to node i compared to all other neighbours of i.

3.4 Method

The key idea of MGSR is using motif attention mechanism to combine multiple motif adjacency matrices in graph convolution. We calculate node's motif attention weights with respect to each motif adjacency matrix and use the weights to calculate the layer output that is the weighted mean of node embeddings obtained from different motif adjacency matrices. The model learns different motif's weight for each node through training so the redundancy in motif adjacency matrices is minimised.

As we use a bipartite graph to represent the mashup-API interaction network, the embeddings of mashup and API nodes have the same dimensions to enable the message-passing in graph convolution layers. In the graph convolution layers, we denote the both mashup and API node embeddings as $h \in \mathbb{R}^d$. In the collaborative filtering part, we use h_m for mashup embeddings and h_a for API embeddings.

The core of our graph convolution layer is the motif attention mechanism, which can be described in node level using the following equations:

$$h_i^{l+1} = \sigma\Big(\sum_{m=1}^{M} \beta_{im}^l W^l h_{im}^l\Big) \tag{3.1}$$

$$\beta_{im}^l = \mathrm{softmax_i}\Big(\mathrm{LeakyReLU}\left(\vec{a}^T[Wh_i^l \| Wh_{im}^l]\right)\Big) \tag{3.2}$$

$$h_{im}^l = \sum_{j \in \mathcal{N}_m(i)} \alpha_{ij} W^l h_j^l \tag{3.3}$$

where $\mathcal{N}_m(i)$ denotes the motif neighbours of node i for motif m, $\alpha_{ij,m}^l$ denotes the attention weight between motif neighbours i, j for motif m calculated using 3.1, h_{im}^l

Table 3.1: Baseline comparison over different p on HR metric

p	2	3	4	5	6	7	8	9	10
AMF	0.4021	0.4152	0.4182	0.4262	0.4352	0.4521	0.4921	0.5201	0.5583
NGCF	0.4372	0.4552	0.5037	0.5537	0.6012	0.6252	0.6317	0.6677	0.6882
MISR	0.6494	0.6760	0.7023	0.7211	0.7286	0.7392	0.7392	0.7407	0.7441
HACF	**0.6645**	**0.7190**	0.7367	0.7483	0.7547	0.7603	0.7604	0.7606	0.7616
GAT-CF	0.5261	0.6176	0.6716	0.7271	0.7647	0.7680	0.7761	0.7761	0.7761
MGSR	0.5976	0.6829	**0.7561**	**0.7805**	**0.8171**	**0.8293**	**0.8293**	**0.8415**	**0.8415**

Table 3.2: Baseline comparison over different p on NDCG metric

p	2	3	4	5	6	7	8	9	10
AMF	0.3321	0.3498	0.3694	0.3972	0.3999	0.4093	0.4109	0.4271	0.4306
NGCF	0.3521	0.3841	0.4393	0.4647	0.4678	0.4580	0.4603	0.4613	0.4647
MISR	0.5209	0.5050	0.5343	0.5301	0.5351	0.5433	0.5569	0.5568	0.5583
HACF	**0.5626**	**0.5666**	0.5627	0.5657	0.5711	0.5716	0.5718	0.5725	0.5726
GAT-CF	0.4857	0.5290	0.5541	0.5714	0.5843	0.5833	0.5857	0.5834	0.5809
MGSR	0.5255	0.5665	**0.6017**	**0.6103**	**0.6238**	**0.6271**	**0.6271**	**0.6311**	**0.6256**

denotes node i's aggregated embeddings from motif neighbours for motif m at layer l, β_{im}^{l} is the motif attention weight of motif m for node i at layer l.

Similar to self-attention, we calculate the dot product between the input embeddings and embeddings generated from different motif adjacency matrices to measure the importance of different motifs and combine them with the attention weights learned. This makes our model aware of the local topological structure during propagation and results in a more efficient message-passing process than GAT. Depending on the type of motif, the motif adjacency matrix may contain one-hop, two-hop or even three-hop connections so nodes can obtain information from more nodes in a single layer. Additionally, each motif represents a type of relationship between nodes in the motif. Separately calculating node embeddings with different motif adjacency matrices can maximise the variance of each motif to learn the most important relationships and eliminate the noises. The motif attention mechanism essentially learns a weighted mean over each node's motif neighbours, which can be seen as a two-dimensional neighbours

set comparing to the one-dimensional neighbours in GAT's attention mechanism.

After obtaining the node embeddings, we split the embedding matrix into two for mashups and APIs to predict invocation between them. The estimated invocation probability $\hat{r}_{ma}$ of a mashup m invoking an API a is the dot product of mashup embedding h_m and API embedding h_a:

$$\hat{r}_{ma} = h_m \cdot h_a \tag{3.4}$$

To train the model parameters, we optimise the binary cross entropy loss which has been commonly used in recommender systems(X. He, Liao et al., 2017):

$$Loss = -\sum_{m \epsilon U, a \epsilon V} r_{ma} \log \hat{r}_{ma} + (1 - r_{ma}) \log (1 - \hat{r}_{ma}) \tag{3.5}$$

where r_{ma} denotes the ground truth of the interaction between mashup m and API a. $r_{ma} = 1$ if m invokes a and 0 otherwise.

3.5 Experimental Evaluation

We tested our model on ProgrammableWeb dataset[1]. The dataset consists of 17829 APIs, 6340 mash-ups, and their invocation data. Specifically, if mashup m invokes API a then there exists an invocation record of 1. Before the invocation data are used to construct the mashup-API graph, we remove all empty nodes and have 5691 mashups and 1170 APIs left. The invocations are split into a train set and a test set with 80% and 20% of all records respectively. We compare our proposed MGSR with the following state-of-the-art approaches:

- AMF(Nguyen, Yu, Nguyen & Han, 2021): An attentional probabilistic matrix factorization model which learns attention scores of mashups and APIs via a

[1]The dataset is crawled from www.programmableweb.com in October 2018.

neural attentional network and combines with context similarity to enhance matrix factorization.

- NGCF(X. Wang, He, Wang et al., 2019): A GCN-based recommendation framework that exploits user-item graph structure by conducting the embedding process on the user-item graphs. Through multiple graph convolution layers it injects the collaborative signal with high-order connectivities in user-item graphs into the embedding process.

- MISR(Ma et al., 2021): A multiplex invocation-oriented service recommendation model which leverages three types of invocations between services and mashups and incorporates them into a DNN to present for their explicit and implicit relationships.

- HACF(Nguyen et al., 2022): An autoencoder based recommendation framework with higher-order data augmentation for collaborative filtering.

- GAT-CF: Our implementation of GAT(Veličković et al., 2018) on collaborative filtering task.

Following previous research(X. He, Liao et al., 2017), we use hit ratio (HR) and normalised discounted cumulative gain (NDCG) to evaluate the performance of the recommended API list. To calculate HR, we first create a list of unused APIs for each mashup in the train set and sort them ascending by their estimated invocation probability. Then we can obtain the top-p HR using the following equation:

$$\text{HR@p} = \frac{\text{Number of hits } @p}{N_r} \tag{3.6}$$

where we denote N_r as the number of APIs to recommend. The NDCG is used to evaluate the precision of recommendation in terms of the rankings of APIs in the

recommendation list. The NDCG from position 1 to p in the recommendation list is calculated using the following equation:

$$\text{NDCG@p} = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(i+1)} \tag{3.7}$$

where rel_i is the graded relevance of the result at position i.

For the evaluation process, all the mashups in the test set have a list of invoked APIs. For each mashup, all APIs in the test set are scored with their estimated invocation probabilities within the range of $(0, 1)$. After that, we sort them ascending to their estimated ratings. Following the recent approaches(X. He, Liao et al., 2017; X. Wang, He, Wang et al., 2019), we truncate the recommendation list at p for both evaluation metrics. Our experiments cover p from 2 to 10.

We implement the proposed methods in Pytorch. The embedding size is fixed to 8. The learning rate is tuned amongst {0.0001, 0.0005, 0.001, 0.005}. The dropout rate is searched in {0.0, 0.1, ..., 0.8}. The model prameters are initialised using Xavier initialiser(Glorot & Bengio, 2010). All models are trained for 1000 epochs for convergence.

Table 3.1 and Table 3.2 show the HR and NDCG results of baseline methods and MGSR over different p. Overall, the proposed model outperforms baseline methods especially when p is larger than 2. The five baseline methods can be categorised from the perspective of graph learning: NGCF, HACF and GAT-CF are graph-based methods while AMF and MISR are not. NGCF do consider the data as a graph and use GCN to learn the high-order information, but GCN is an isotropic model, which means it ignores the different importance of neighbours and leads to over-smoothing issue. In comparison, both HACF and GAT-CF consider the different weights between neighbours so that the high-order neighbourhood aggregation is less affected by over-smoothing. Although MISR ignores the information within graph structure, it includes

extra information of multiple types of connectivity between nodes to improve their performance. Our MGSR model achieves significantly better results without the need of extra data. By exploiting the network motifs in the graph, MGSR captures the local topological structure which implies the difference between different neighbours. By injecting the motif information into node embeddings, our model needs less layers to propagate the high-order information in the graph.

3.6 Conclusion and future work

In this work, we study the network motifs for bipartite graphs and accordingly propose a new framework MGSR, which achieves superior performance by exploiting higher-order information in motifs. The model also uses a motif attention mechanism to minimise the redundancy between different types of motifs, alleviate the issue of over-smoothing. This work represents an initial attempt to leverage motifs in graph representation learning for collaborative filtering. There are different techniques that can be incorporated with motif analysis such as dimension reduction methods and spectral analysis which will be explored in our future work.

Chapter 4

Motif-Based Graph Convolutional Networks for Recommendation

4.1 Introduction

Graph neural networks (GNNs) have emerged as a promising solution for multiple industries such as e-Commerce (Piao et al., 2021; Z. Wang et al., 2020; W. Yu & Qin, 2020; X. Zhou et al., 2019) and social media (Anwaar et al., 2021; X. He et al., 2020; Q. Wu et al., 2019) in favour of their superior performance in graph learning tasks including node classification, graph classification, and link prediction. A series of recent advancements in recommender systems (X. Wang, He, Wang et al., 2019; X. He et al., 2020; W. Yu et al., 2022) that adopt Graph Convolutional Network (GCN) (T. N. Kipf & Welling, 2017) show great potential of graph-based Collaborative Filtering (CF). Graph-based CF represents users and items as nodes and interactions as edges within a graph structure, which allows for the exploitation of higher-order connections. However, there are two key challenges for graph-based CF methods. One is the data sparsity issue in user-item interactions, which makes it more challenging to generate reliable recommendations. The other challenge is the lack of local pattern exploitation in the

existing graph-based CF methods.

Although GNNs can learn from higher-order connections, it relies on multi-layer propagation. Local structural patterns cannot be captured directly by normal graph convolution. Therefore, our rationale is to investigate the effectiveness of capturing such local patterns for graph-based collaborative filtering to enhance the model's learning ability at each layer. This achieves similar effects to additionally preserving intermediate layer embeddings into the output layer as lower-order and higher-order interactions are both captured in the final embeddings (K. Xu et al., 2018).

As a simple and common higher-order patterns in graphs, network motifs have shown promising enhancement to graph representation learning on homogeneous graphs such as social networks (Monti et al., 2018; H. Zhao et al., 2021; Piao et al., 2021; X. Li, Wei, Feng, Liu & Zheng, 2021) and bioinformatics networks (H. Peng et al., 2020; X. Chen et al., 2022), as well as heterogeneous networks (Dareddy, Das & Yang, 2019; X. Liu & Song, 2022; Z. Yu & Gao, 2022; Q. Hu, Lin, Wang & Li, 2022). Formally, we define network motifs as recurring patterns or subgraphs that appear in graph networks more frequently than in randomised graphs (Milo et al., 2002). They can be considered the fundamental building blocks of graphs, capturing local structures that are often indicative of underlying node behaviours and characteristics. If multiple nodes can form a motif instance, they are considered as motif neighbours and are connected in the motif adjacency matrix, even if some of the nodes are not really connected in the graph. This means motif adjacency matrices can be used to preserve higher-order neighbours' information in a single layer's graph convolution operation.

By incorporating motifs in layer-wise propagation, nodes learn the local structure so more information can be captured than using the original adjacency matrix. In addition, network motifs can mitigate the limitation of user-item interaction data sparsity by utilising local connectivity patterns to infer missing links, which then enriches the data for recommendations. By identifying and applying motifs to a GNN framework, we

can gain deeper insights into the structural properties of any given user-item interaction graphs, which can lead to more accurate and robust recommendation outcomes. Based on the rationales discussed above, we propose a new graph neural network that preserves higher-order information in each single-layer neighbourhood aggregation using network motifs.

In Chapter 3, we propose a motif-based graph attention network for service recommendation (MGSR). The model uses a motif attention mechanism that first calculates graph attentions on multiple motif adjacency matrices and then combine the outputs using a motif attention which measures the similarity between previous layer's hidden states and the outputs from the motif adjacency matrices. This means the motif attention mechanism with M motifs will have a significantly larger computational complexity than the original graph attention network (Veličković et al., 2018), which is $O\Big(M \cdot K \cdot (|V| \cdot d \cdot d' + |E| \cdot d')\Big)$ where M is the number of motifs, K is the number of attention heads, d is the input feature dimension and d' is the output feature dimension. Additionally, due to the nature of motif adjacency matrix, the matrix has a higher density than the original adjacency matrix so E is significantly larger than the actual number of edges.

While MGSR is suitable for relatively small datasets like a mashup-API dataset in service computing, it is challenging to scale to large datasets used in general recommender systems, such as the Amazon datasets. To overcome this limitation and explore the impact of motifs in general recommender systems, we propose a series of efficient motif adjacency matrix generation algorithms and a generic framework for recommender systems called MotifGCN.

MotifGCN integrates motifs in graph convolution layers to enhance the neighbourhood aggregation efficiency within each single layer. Compared to MGSR, MotifGCN does not use any form of attention mechanism and has fewer learnable parameters hence

has a significantly improved computational complexity of $O\Big(|V| \cdot d\Big)$. Our experiments on four real-world recommendation datasets show that motifs can significantly improve GCNs for CF.

Our contributions are summarised as follows:

- We propose a novel CF framework called MotifGCN, which integrates motifs in graph convolution layers to enhance the neighbourhood aggregation efficiency within each single layer. This generic framework can be applied to most recommendation datasets.
- We propose dedicated algorithms of generating motif adjacency matrices for all motifs consisting of up to four nodes in bipartite graphs. Theoretical analyses and experiments show that the algorithms have improved time and space complexity over general approaches.
- We conduct experiments on four real-world datasets to demonstrate the effectiveness of our proposed framework.

The rest of this chapter is organised as follows. Section 4.3 provides the background knowledge to understand our proposed method. Section 4.4 presents our proposed motif adjacency matrices generation algorithms and MotifGCN architecture. Section 4.5 shows our experimental results and performance analysis. Section 4.6 discusses how motif adjacency matrix changes graph convolution and finally Section 4.7 concludes this chapter.

4.2 Related Work

In this paper, our work is dedicated to motif-based GNN for CF. To the best of our knowledge, we are the first to explore the application of network motifs specifically

on bipartite graphs within this context. For this section, we extend the scope slightly and review current literature in the fields of GNNs for CF, motif-based GNNs and higher-order enhanced GNNs for CF in Sections 2.1, 2.2 and 2.3 respectively.

4.2.1 Graph Neural Networks for Collaborative Filtering

There has been an extensive collection of research works on GNNs for CF in recent couple of years. Following the earlier works of GraphSAGE (W. Hamilton et al., 2017) and GCN (T. N. Kipf & Welling, 2017), Neural Graph Collaborative Filtering (NGCF) (X. Wang, He, Wang et al., 2019) is one of the earliest works to apply GCN to CF. The paper proposes a message-passing design for convolution layers with a short circuit mechanism that concatenates the hidden states after each layer as the final output. The datasets used are represented as bipartite graphs so edges only exist between user-item pairs. The message between a user-item pair is the weighted average of the item node embedding and the Hadamard product of the two nodes' embeddings. The convolution layers then aggregate the messages for all nodes to update their embeddings. After obtaining the final node embeddings, the prediction of the interaction between a user-item pair is calculated via the inner product of their embeddings.

Influenced by NGCF, He et al. (X. He et al., 2020) perform ablation studies on NGCF and show that feature transformation matrix and nonlinear activation function do not have additional effectiveness for GCNs on CF tasks. In the same work, an NGCF-inspired model called LightGCN is proposed to eliminate the feature transformation matrix and nonlinear activation function in the convolution layers. In a way, LightGCN can be considered a reduced and refined version of the NGCF model while further enhancing experimental performances.

As another key publication, Yu et al. (W. Yu & Qin, 2020) propose Low-pass Collaborative Filter Network (LCFN) that simplifies spectral graph convolution by

calculating only a small number of graph Laplacian eigenvectors of small eigenvalues. The work shows that small eigenvalues correspond to low-frequency graph signals which can reflect nodes being connected, while large eigenvalues correspond to high-frequency graph signals which represent noises in data. Through removing high-frequency eigenvectors in graph Laplacian, LCFN avoids high computational cost of eigen-decomposition and eliminates noise in graphs, thereby improves model accuracy and efficiency.

LCFN has a limitation of constructing separate graphs for user nodes and item nodes thus ignores the direct interaction between a user and an item. To address this issue, Low-pass Graph Convolutional Network (LGCN) (W. Yu et al., 2022) uses bipartite graphs to preserve the user-item interactions. Similar to LightGCN, LGCN also simplifies the graph convolution layer by removing feature transformation and nonlinear activation. The authors also show that low-pass filter enables LGCN to avoid over-smoothing issue.

Ouyang et al. (Ouyang et al., 2024) identify that traditional CF models tend to overfit due to noise in labelled training data and propose to mitigate this issue by integrating contrastive learning with a GNN to enhance the alignment and uniformity of user and item representations independently of labels. Alignment is improved by minimizing the distance between representations of interacting user-item pairs. Uniformity is enhanced by distributing these representations more evenly across their respective hyperspheres. Their framework also leverages a novel 0-layer embedding perturbation mechanism for minimal data augmentation, allowing the model to learn these properties in a label-independent manner with faster convergence.

4.2.2 Motif-based Graph Neural Networks

A series of recent works show that the topological structure of graphs has significant influence on local and global representation learning (K. Xu et al., 2019; Bouritsas et al., 2022). As one way of exploiting the topological information in graphs, network motifs have been used in a range of research works for GNNs. In this section, a range of motif-based GNNs will be explained.

As one of the first works using motifs in GNNs, MotifNet (Monti et al., 2018) uses motifs for directed graphs. The work obtains motif adjacency matrices from directed motifs and then get the motif Laplacian. Although the adjacency matrices are still symmetric, different motifs induce different structures which enables the model to be anisotropic. For the core structure of MotifNet, multivariate polynomials of degree p are applied to the motif Laplacians. Due to the large number of coefficients required, the polynomials are simplified using two approaches. One is to only consider the two motifs of incoming and outgoing edges. The other is to define the polynomials with recursive products of each degree.

Rossi et al. (R. A. Rossi et al., 2018) propose a framework that uses motifs to learn higher-order network embeddings which represent interactions between multi-hop neighbours. They adopt the same definition of motif adjacency matrices as in (Benson et al., 2016) except that they use undirected motifs. Additionally, the authors propose k-step motif adjacency matrix which is the power of the original motif adjacency matrix. For m motifs with k steps, there are $m \times k$ motif adjacency matrices in total. This embedding method is adopted in Motif Convolutional Networks (J. B. Lee et al., 2019) which uses an attention mechanism and an epsilon greedy strategy to select only one neighbour for each node's neighbourhood aggregation.

With a focus on heterogeneous graphs, Motif-CNN (Sankar et al., 2019) also uses motifs to capture higher-order information. The authors differentiate nodes into different

semantic roles according to their types and occurrences in the motifs. Instead of the adjacency matrix in GCN, Motif-CNN uses a motif adjacency matrix weighted by semantic roles to perform convolution operation. Nodes with the same semantic role in all instances of a motif share the same weight. For the weights of different motifs, Motif-CNN adopts the attention mechanism to dynamically adjust the importance of different motifs for each node. Similarly, InfoMotif (Sankar et al., 2020) also captures multi-hop interactions using structural roles of nodes in motifs. Nodes are considered as the same attributed role if they have co-varying attributes in similar motif instances. To identify structurally similar nodes, InfoMotif adopts mutual information maximisation to obtain attribute correlations in motifs.

Li et al. (H. Li et al., 2020) propose to incorporate global information with motif-based Subgraph Convolutional Neural Networks. Instead of directly linked neighbours, motif neighbours are used for neighbourhood aggregation. An input graph is transformed to an m-ary tree by cascading each node's m-ary subtree following the order of node degrees. The m-ary tree is then used to generate an adjacency matrix for feature aggregation.

The recent works of Motif-based Multiview Graph Attention Networks with Gated Fusion (Piao et al., 2021) and Motif Graph Neural Network (X. Chen et al., 2022) have rather similar structures. They both follow the commonly used message-passing paradigm of GNN framework. Specifically, they first generate motif adjacency matrices and then aggregate node features on each motif adjacency matrix. The key difference between the two pieces of work is how the features aggregated from different motif adjacency matrices are combined. The former uses an attention layer to get a weighted average from the feature matrices before cascading a gated fusion layer to control the weights of residual connections. The latter minimises redundant information among all motifs. Here, the redundant information is measured by the similarity between node features aggregated from each motif adjacency matrix and those from all other motif

adjacency matrices. An inner product and a non-linear activation function are used in the similarity calculation.

Motifs are also used in graph classification. Zhang et al. (S. Zhang et al., 2024) present a novel framework that uses motifs in GNNs pre-training for molecular property prediction. The framework learns subgraph embeddings by maximising the likelihood of each subgraph belonging to a motif, and then optimise the embeddings by combining them to contrast with initial who-graph embeddings. Similarly, Zhou et al. (Z. Zhou, Zhou et al., 2024) also use motifs to learn subgraph embeddings for graph classification. Differently, they rank subgraph importance by calculating the similarity between subgraph embeddings and a graph-level vector, and combine important subgraphs into the overall graph embedding. In the subgraph-motif alignment phase, top candidate subgraphs are selected based on importance scores, and motifs are iteratively extracted using k-means clustering. These motifs guide the alignment of subgraph representations through a contrastive loss function, ensuring that subgraphs align with motifs of their class while remaining distinct from others.

Despite the rich collection of motif applications in GNNs, there is currently very limited research in motif-based CF. This gap motivates our proposed MotifGCN, which embeds motifs directly into graph convolution to achieve efficient single-layer higher-order aggregation with improved adjacency construction for all bipartite motifs up to four nodes. Together, these chapters provide CF-specific motif building blocks that address missing structure, targeted aggregation, and computational practicality.

4.2.3 Higher-order Enhanced Graph Neural Networks for Collaborative Filtering

Although there has been considerable amount of research on motif-based GNNs, applications of motif on CF task are still limited. Here we review some CF models that

consider higher-order structure in their architecture designs, which is also related to our work as motif is a tool for capturing higher-order connectivity as well.

HyperRec (J. Wang et al., 2020) is a recommendation model that employs hypergraphs to represent higher-order relationships between users and items. The hypergraph is constructed such that each hyperedge represents a group of items that a user has interacted with within a specific time window, thereby encapsulating the sequential and co-occurrence information in the user's interaction history. By utilizing sequential hypergraphs, the model captures complex dependencies between short-term user intent and dynamic item embeddings, offering a richer and more comprehensive user representation compared to traditional user-item graphs.

Ji et al. (Ji et al., 2020) propose a model called Dual Channel Hypergraph Collaborative Filtering which also leverages hypergraphs to capture higher-order relationships among users and items, but form two hypergraphs for user-user and item-item respectively. The hyperedge in user-user hypergraph is a group of users who share common interactions with one or more items. Similarly, the hyperedge in item-item hypergraph is a group of items that are connected with at least one same user. Through the two hypergraphs, the model can learn complex higher-order dependencies within users and items.

Zhang and McAuley (H. Zhang & McAuley, 2020) introduce a model that mixes multiple order's higher-order graph convolutions within a stacked structure to capture a broader range of user-item interactions. This hybrid approach is an adaptation of the shortcut technique (K. He et al., 2015; K. Xu et al., 2018) as it essentially concatenates interim representations to generate the final embeddings. Similarly, Nguyen et al. (Nguyen et al., 2022) incorporate higher-order connections within autoencoder framework by augmenting the interaction graph with second-order and fourth-order interactions.

Yu et al. (J. Yu et al., 2021) utilise multi-channel hypergraphs with self-supervised

learning strategy. The channels are user views generated from different aspects such as social relationships and common purchases. Auxiliary tasks are then generated from these channels for self-supervised learning. Hypergraph Contrastive Collaborative Filtering (L. Xia et al., 2022) also combine hypergraphs and self-supervised learning for CF. Specifically, hypergraph is constructed based on mutual connections for both users and items. Both the hypergraph and original interaction graph are used to generate contrastive views by edge dropout. The contrastive loss for self-supervised learning maximizes the agreement between same nodes in different views and minimizes the agreement between different nodes. This objective helps the model to refine the embeddings, ensuring that similar nodes have closer representations, while dissimilar ones are pushed apart.

Xia et al. (J. Xia et al., 2024) introduce a graph signal processing approach that uses two types of filters to separately learn user unique behaviours and global behaviours. The user unique behaviours are learned among users that share similar preferences, preventing user-specific preferences from being influenced by the behaviours of dissimilar users. The global behaviours are general interaction patterns across all users, which are captured by a combination of higher-order and ideal low-pass filters. This balances the need to leverage broad user interaction data while avoiding the over-smoothing problem commonly associated with traditional filtering methods.

Gong et al. (Gong et al., 2024) design two higher-order neighbour-enhanced strategies to improve GNNs for CF. The embedding propagation (EP)-based strategy and the positive user–item pair (PP)-based strategy. The EP-based strategy employs a novel higher-order graph convolution method that mitigates the excessive decay of higher-order neighbour information by assigning different weight coefficients to various order neighbours, thereby capturing richer collaborative signals. The PP-based strategy enhances the model's training process by identifying potentially positive samples from a user's higher-order neighbours, based on item similarity and user similarity, to improve

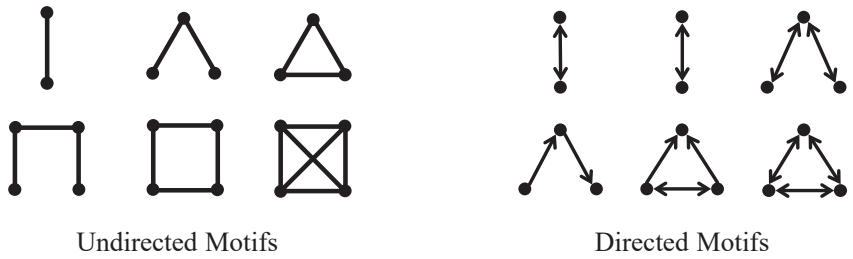


Figure 4.1: Examples of undirected motifs and directed motifs

the quality of the positive user–item pairs used during model training.

4.3 Background

In this section, we first introduce network motifs and explain the differences between motifs in homogeneous graphs and bipartite graphs. We then discuss the difference and relationship between motif adjacency matrices and powers of adjacency matrix.

4.3.1 Network Motifs in Bipartite Graphs

Network motifs are recurring and significant patterns or subgraphs that appear in complex networks more frequently than in randomized networks (Milo et al., 2002). As most motifs involve more than two nodes, they can provide useful local structure information and also contribute to global properties of the whole graph. Depending on the type of graphs, motifs can be either directed or undirected. Figure 4.1 shows some common motifs in directed and undirected graphs. Our proposed method uses bipartite graphs to represent user-item interactions. This is a natural choice for collaborative filtering as bipartite graphs can effectively represent the relationships between the two distinct user set and item set.

We provide the formal definition of bipartite graphs as follows.

Definition 1 (Bipartite graphs (Diestel, 2017))**.** *A bipartite graph $G = (U, V, E)$ is a graph where all nodes in G are partitioned into two disjoint classes U and V, i.e. $U \cap V = \varnothing$, such that every edge has its terminal node in different classes from its initial node: nodes in the same partition class must not be adjacent.*

The interactions between users and items are usually not symmetric in recommender systems, e.g. in movie rating a rating is established by a user for a movie but a movie cannot establish an interaction with a user. This means there does not exist a symmetric pair of directed edges between any two nodes. To define graphs for recommender systems, we need to define oriented graphs first.

Definition 2 (Oriented graphs (West & others, 2001))**.** *An oriented graph $G = (V, E)$ is a directed graph such that none of its node pairs is linked by more than one edge and none of its edges has the same initial and terminal node.*

Now we can define oriented bipartite graphs, which can be used to represent recommendation data.

Definition 3 (Oriented bipartite graphs (Das, Ghosh, Ghosh & Sen, 2021))**.** *An oriented bipartite graph $G = (U, V, E)$ is a directed bipartite graph such that for any $u \in U, v \in V, uv \in E$ implies $vu \notin E$.*

We can use oriented bipartite graphs to represent recommender systems datasets but finding instances of directed motifs in oriented bipartite graphs is in fact equivalent to identifying undirected motifs in undirected bipartite graphs. The generated motif

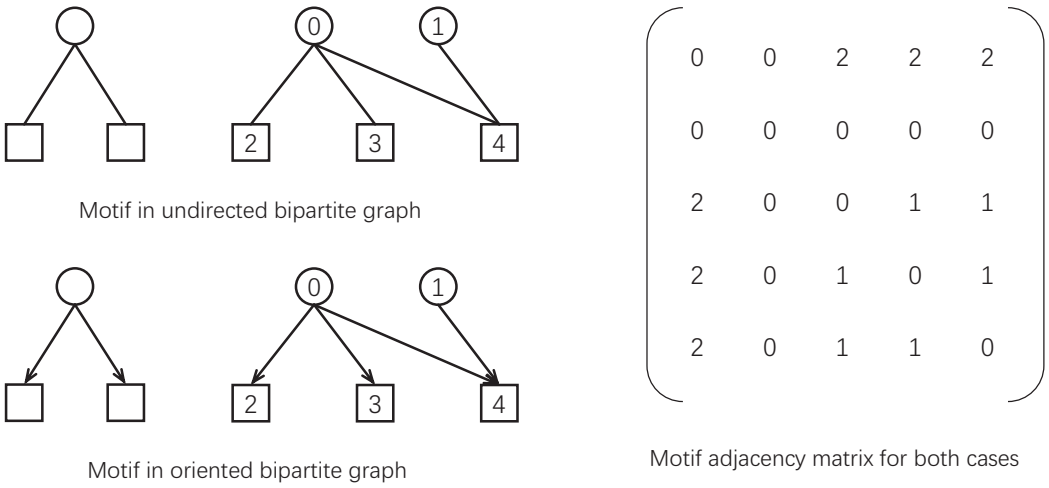


Figure 4.2: Example motif in undirected bipartite graph and oriented bipartite graph. Circle nodes and square nodes are used to represent two types of nodes, e.g. users and items.

adjacency matrices are the same because they only record the coexistence of nodes in motif instances. An example of this equivalence is shown in Figure 4.2. For the ease of understanding and implementation, we consider the graphs and motifs as undirected in this work. If the datasets are represented as directed graphs, our methods are completely compatible and no further modification is needed.

Due to the absence of user-user edges and item-item edges, the complex types of motifs in homogeneous graphs such as four-clique does not exist in bipartite graphs. This makes bipartite graphs not as computationally expensive as homogeneous graphs for motif analysis. For the purpose of exploring different motifs within limited computational resources, we only consider motifs consisting of up to four nodes. Figure 4.3 shows all seven motifs used in this work.

4.3.2 Motif Adjacency Matrix

A common approach to incorporate network motifs into existing graph learning frameworks is to form motif adjacency matrices (Benson et al., 2016). Given a graph and a

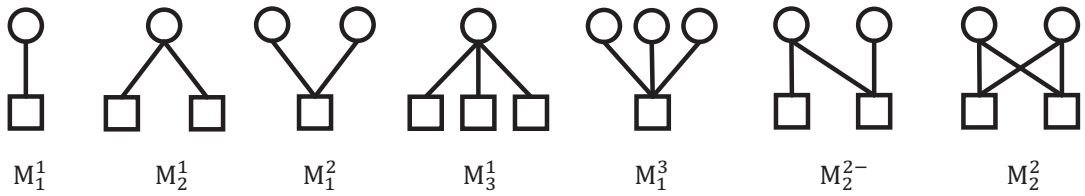


Figure 4.3: All motifs consisting of two to four nodes in bipartite graphs. Superscript refers to the number of user nodes in the motif while subscript means the number of item nodes in the motif. The hyphen in M_2^{2-} indicates one less edge in the motif compared to M_2^2.

certain motif m, we can form the motif adjacency matrix $\mathbf{A}_m$ whose elements (i, j) are the number of instances that nodes i and j co-exist in. We refer to nodes that co-exist in at least one motif instance as motif neighbours. A common approach to generate a motif adjacency matrix is to carry out a subgraph isomorphism test between the motif and every subgraph in the graph. An example is the VF2 (Cordella, Foggia, Sansone & Vento, 2004) graph matching algorithm, which has a complexity of $O(N!N)$ where N is the number of nodes in the graph. To improve the computational efficiency, we introduce our dedicated algorithm for generating motif adjacency matrices in Section 3.1.

4.4 Method

In this section, we first present the algorithms for generating motif adjacency matrices for motifs in bipartite graphs and then introduce the architecture of our proposed Motif Graph Convolutional Network.

4.4.1 Generating Motif Adjacency Matrices for Bipartite Graphs

Previously, Bouritsas et al. (Bouritsas et al., 2022) have tried counting motifs using the generic subgraph isomorphism algorithm VF2 (Cordella et al., 2004). The method is usable for small graphs such as the Cora dataset, but cannot cope with large graphs such

Algorithm 1: Motif adjacency matrix generation algorithm for M_2^1.

Input: Edges E; number of nodes N; adjacency matrix A; neighbourhood function $\mathcal{N}$
Output: Motif adjacency matrix $\mathbf{A}_m$

$d \leftarrow \{deg(i) - 1 | \forall i \in U\}$;
$D \leftarrow N \times N$ diagonal matrix extended from d;
$M \leftarrow D \cdot A$;
for $i \in U$ **do**
  if $|\mathcal{N}(i)| > 1$ **then**
    for $(j, k) \in \binom{|\mathcal{N}(i)|}{2}$ *pairs of APIs* **do**
      Add one to entry (j, k) in M;
    end
  end
end
$M \leftarrow M + M^T$;

as the Amazon-Book dataset due to a high complexity of $O(N!N)$. As most datasets for recommender systems have a very low density, we decide to design tailored algorithms to generate motif adjacency matrices for each motif. As shown in Figure 4.3, motif M_1^1 is simply an edge between two nodes, so the motif adjacency matrix is the same as the original adjacency matrix and we follow the generation method used in GCN (T. N. Kipf & Welling, 2017).

Algorithm 1 describes the motif adjacency matrix generation process for motif M_2 in which all edges E, the number of nodes N in the graph, and the graph's adjacency matrix A are provided as input. The algorithm begins with splitting all edges into groups. In each group Q_i, all edges source from the same mashup node i. We then generate a vector where the elements are the node degrees minus one, i.e.$deg(i) - 1$. After that, we assign zero to the counts for API nodes in d because only degrees for mashup nodes are used for update the motif adjacency matrix. Next, we generate an upper triangular matrix from multiplying the adjacency matrix A with a diagonal matrix built from d. In each step of the outer loop, we first check whether the number of edges in group Q_i is greater than one since there has to be at least two API nodes plus one

Algorithm 2: Motif adjacency matrix generation algorithm for M_2^{2-}.

Input: Edges E; number of nodes N; neighbourhood function $\mathcal{N}$; intersection function $INTERSECT$; symmetric difference function $SYMDIFF$
Output: Motif adjacency matrix $\mathbf{A}_m$

1 $\mathbf{A}_m \leftarrow N \times N$ matrix filled with 0;
2 **for** $\forall (j,k) \in V$ **do**
3 $\quad S \leftarrow INTERSECT(\mathcal{N}(j), \mathcal{N}(k))$;
4 $\quad P \leftarrow SYMDIFF(\mathcal{N}(j), \mathcal{N}(k))$;
5 $\quad n_s \leftarrow |S|$;
6 $\quad n_p \leftarrow |P|$;
7 $\quad$ **if** $n_s > 0$ *and* $n_p > 0$ **then**
8 $\quad\quad$ Add $n_s \times n_p$ to element (j,k) in $\mathbf{A}_m$;
9 $\quad\quad$ **for** $\forall i \in S$ **do**
10 $\quad\quad\quad$ Add n_p to elements (i,j) and (i,k) in $\mathbf{A}_m$;
11 $\quad\quad\quad$ **for** $\forall l \in P$ **do**
12 $\quad\quad\quad\quad$ Add 1 to element (i,l);
13 $\quad\quad\quad$ **end**
14 $\quad\quad$ **end**
15 $\quad\quad$ **for** $\forall l \in P$ **do**
16 $\quad\quad\quad$ Add n_s to elements (l,j) and (l,k) in $\mathbf{A}_m$;
17 $\quad\quad$ **end**
18 $\quad$ **end**
19 **end**
20 $\mathbf{A}_m \leftarrow \mathbf{A}_m + \mathbf{A}_m^T$

mashup node to form an instance of $M1_2$. If so, it enters the inner loop which adds one to the counts between every pair of API nodes because any two API nodes can only coexist in one motif instance of $M1_2$. At the end of this algorithm, we add matrix M's transpose matrix to itself to form a symmetric matrix.

The algorithms for motif M_1^2, M_3^1, M_1^3 are very similar to 1, so we omit their algorithms and provide steps to alter the algorithm: (1) For M_1^2, simply replace all appearances of U with V. (2) For M_3^1, replace $(j,k) \in \binom{|\mathcal{N}(i)|}{2}$ at line 6 with $(j,k,l) \in \binom{|\mathcal{N}(i)|}{3}$ and follow the same steps of line 7 for (j,l) and (k,l); (3) For M_1^3, refer to both (1) and (2).

The algorithms for motif M_2^{2-} and M_2^2 are shown as follows.

Algorithm 2 details the method for constructing the motif adjacency matrix for

motif M_2^{2-}. The process begins by grouping all edges according to their associated item nodes, and an empty matrix $\mathbf{A}_m$ is initialized. In each iteration of the outer loop, with j and k representing any two item nodes, the algorithm calculates both the intersection S and the symmetric difference P of the neighbours of j and k. The sizes of S and P are denoted by n_s and n_p respectively. If S has at least one node and P has at least one node, the algorithm proceeds to update four types of counts: 1) The count between j and k is increased by$n_s \times n_p$, since a motif instance requires one mutual neighbour and one non-mutual neighbour of j and k. 2) For each mutual neighbour i, the counts between i, j and i, k are increased by n_p, reflecting the number of ways to choose one node from P. 3) For each $i \in S$ and $l \in P$, the count between i and l is incremented by one, as i, l, j, and k can form only one motif instance. 4) For each node $l \in P$, the counts between l, j and l, k are increased by n_s, corresponding to the number of ways to select one node from S. The algorithm concludes by adding the transpose of $\mathbf{A}_m$ to itself to produce a symmetric matrix.

Algorithm 3 outlines the procedure for generating the motif adjacency matrix for motif M_2^2. Initially, all edges are grouped based on common item nodes, and an empty matrix $\mathbf{A}_m$ is established. In each iteration of the outer loop, where j and k represent any two item nodes, the algorithm identifies the intersection S of the neighbours of j and k. If S contains at least two nodes, three types of counts are updated: 1) The count between j and k is increased by $\frac{n(n-1)}{2}$, since forming an instance of M_2^2 requires selecting any two of the n users in S, which has $\binom{n}{2}$ possibilities. 2) For each user node i, the counts between i, j and i, k are increased by $n - 1$, as there are $n - 1$ options for choosing the final user node to form an M_2^2 instance with i,j, and k. 3) The count between any two user nodes in S is incremented by one, since any two user nodes can only appear together in one M_2^2 instance with j and k. Finally, to ensure symmetry, the transpose of matrix $\mathbf{A}_m$ is added to itself.

To help better understand the advantage of our proposed algorithms, we present the

Algorithm 3: Motif adjacency matrix generation algorithm for M_2^2.

Input: Edges E; number of nodes N; neighbourhood function $\mathcal{N}$; intersection function $INTERSECT$

Output: Motif adjacency matrix $\mathbf{A}_m$

$\mathbf{A}_m \leftarrow N \times N$ matrix filled with 0;
for $\forall (j,k) \in \binom{|V|}{2}$ *pairs of items* **do**
  $S \leftarrow INTERSECT(\mathcal{N}(j), \mathcal{N}(k))$;
  $n_s \leftarrow |S|$;
  if $n_s > 1$ **then**
    Add $\frac{n_s(n_s-1)}{2}$ to element (j,k) in $\mathbf{A}_m$;
    for $\forall i \in S$ **do**
      Add $n_s - 1$ to elements (i,j) and (i,k) in $\mathbf{A}_m$;
    end
    for $\forall (i,l) \in \binom{n_s}{2}$ *pairs of users* **do**
      Add one to element (i,l) in $\mathbf{A}_m$;
    end
  end
end
$\mathbf{A}_m \leftarrow \mathbf{A}_m + \mathbf{A}_m^T$

average time complexity for all algorithms and a comparison against the VF2 algorithm.

The average time complexity of algorithm for motif M_2^1 is:

$$\begin{aligned} & O(1) + O\Big(|U| \cdot \binom{\frac{|E|}{|U|}}{2}\Big) \\ =& O\Big(|U| \cdot (\frac{|E|}{|U|})^2\Big) \\ =& O(\frac{|E|^2}{|U|}) \end{aligned} \tag{4.1}$$

For sparse graphs, $|E| = c|N|$ so the final complexity for M_2^1 is $O(N)$ which is faster than VF2's best case $O(N^2)$. As the topological structure of M_2^1 and M_1^2 is identical, they have the same average time complexity.

The average time complexity of algorithm for motif M_3^1 is:

$$\begin{aligned} & O(1) + O\Big(|U| \cdot \binom{\frac{|E|}{|U|}}{3}\Big) \\ =& O\Big(|U| \cdot (\frac{|E|}{|U|})^3\Big) \\ =& O(\frac{|E|^3}{|U|^2}) \end{aligned} \tag{4.2}$$

For sparse graphs, $|E| = c|N|$ so the final complexity for M_2^1 is $O(N)$ which is faster than VF2's best case $O(N^2)$. As the topological structure of M_3^1 and M_1^3 is identical, they have the same average time complexity.

The average time complexity of algorithm for motif M_2^{2-} is:

$$\begin{aligned} & O\Big(\binom{|V|}{2} \cdot (|S| + |S| \cdot |P| + |P|)\Big) \\ =& O\Big(|V|^2 \cdot |S| \cdot |P|\Big) \\ \leq& O\Big(|V|^2 \cdot (\frac{|S| + |P|}{2})^2\Big) \\ \leq& O\Big(|V|^2 \cdot (\frac{|E|}{|V|})^2\Big) \\ =& O(|E|^2) \end{aligned} \tag{4.3}$$

For sparse graphs, $|E| = c|N|$ so the final complexity is $O(N^2)$ which is the same as VF2's best case.

The average time complexity of algorithm for motif M_2^2 is:

$$\begin{aligned} & O\Big(\binom{|V|}{2} \cdot (1 + |S| + \binom{|S|}{2})\Big) \\ =& O\Big(|V|^2 \cdot |S|^2\Big) \\ \leq& O\Big(|V|^2 \cdot (\frac{|E|}{|V|})^2\Big) \\ =& O(|E|^2) \end{aligned} \tag{4.4}$$

For sparse graphs, $|E| = c|N|$ so the final complexity is $O(N^2)$ which is the same as

VF2's best case.

4.4.2 Motif-Based Graph Convolutional Network

In this section, we present the architecture of our framework MotifGCN, as illustrated in Figure 4.4. We first generate motif adjacency matrices from the interaction graph and then combine them with a combination weight. The combined matrix will then be sent to graph convolution layers to generate the final embeddings. After that, the user and item embeddings are split and used to calculate the dot product for predicting user's preference. The motif adjacency matrices generation algorithm only needs to run once for each motif on each dataset and the matrices can be reused in later experiments. The node-level propagation of MotifGCN is defined by the following equations.

$$\mathbf{h}_u^{l+1} = \sum_{m \in M} \sum_{i \in \mathcal{N}_m(u)} \frac{1}{c_{ui}} \alpha_m \mathbf{h}_i^l \tag{4.5}$$

$$\mathbf{h}_i^{l+1} = \sum_{m \in M} \sum_{u \in \mathcal{N}_m(i)} \frac{1}{c_{ui}} \alpha_m \mathbf{h}_u^l \tag{4.6}$$

In equations 4.5 and 4.6, $\mathbf{h}_u^l, \mathbf{h}_i^l \in \mathbb{R}^d$ denote the embedding vectors of user node u and item node i at layer l. M is the set of all selected motifs including m_1. $\mathcal{N}_m(u)$ and $\mathcal{N}_m(i)$ denote the motif neighbours of user node u and item node i for motif m respectively. The normalization constant c_{ui} is computed by $\sqrt{|\mathcal{N}_m(u)||\mathcal{N}_m(i)|}$ and α_m is the motif combination coefficient of motif m.

To explore whether network motifs can improve GCNs, we hope to emphasise the influence of motif adjacency matrices to the performance as much as possible. Following LightGCN (X. He et al., 2020), we ignore the linear transformation matrix, non-linear activation function and self-loops in the original GCN layer (T. N. Kipf & Welling, 2017) to restrict the quantity of parameters. To explicitly test the performance of individual motifs, we set $M = \{m_1, m\}$ for each motif m other than m_1 and set

$\alpha_1 = \alpha_x = 0.5$. The motif combination coefficient can also be learned automatically such as using the attention mechanism (Sankar et al., 2019), which will be explored in future work.

After obtaining the final node embeddings, the prediction scores between any user-item pairs can be calculated as the dot product of their embedding vectors:

$$\hat{y}_{ui} = \mathbf{h}_u^T \cdot \mathbf{h}_i \tag{4.7}$$

The prediction scores will be ranked to generate the top-p list of nodes as the recommendation results.

The matrix form of the layer propagation is provided as follows:

$$\mathbf{H}^{l+1} = \widetilde{\mathbf{A}}_{mc}\mathbf{H}^l, \tag{4.8}$$

$$\widetilde{\mathbf{A}}_{mc} = \sum_m \alpha_m \widetilde{\mathbf{A}}_m, \tag{4.9}$$

$$\widetilde{\mathbf{A}}_m = \mathbf{D}_m^{-1/2}\mathbf{A}_m\mathbf{D}_m^{-1/2}, \tag{4.10}$$

where $\widetilde{\mathbf{A}}_{mc}$ is the motif combination matrix, $\widetilde{\mathbf{A}}_m$ is the symmetrically normalised motif adjacency matrix, and $\mathbf{D}_m$ is the degree matrix of motif m's motif adjacency matrix $\mathbf{A}_m$. The motif adjacency matrices including the original adjacency matrix are generated from the edge set E and $\mathbf{A}_m \in \mathbb{R}^{N\times N}$. The embedding matrix at layer l is denoted by $\mathbf{H}^l \in \mathbb{R}^{N\times d}$.

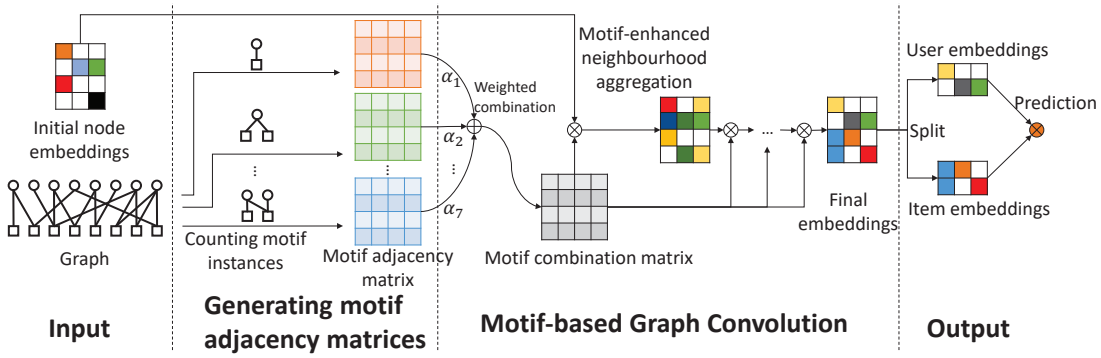


Figure 4.4: An illustration of MotifGCN model architecture.

4.4.3 Optimization settings

Following the common choices of optimisation strategies for collaborative filtering, we use Bayesian Personalised Ranking (Rendle, Freudenthaler, Gantner & Schmidt-Thieme, 2009) as loss function and Adaptive Moment Estimation (Kingma & Ba, 2015) as optimiser. For the generalisation strategy, we adopt L2 regularisation without dropout since our model only has the initial node embeddings as the learnable parameters.

4.5 Experiments

In this section, we present experimental evaluations of our work in two subsections which correspond to sub-RQ1 and sub-RQ2. To help answer the first sub-research question, we evaluate the efficiency of our dedicated motif adjacency matrix generation algorithms. With respect to the second sub-research question, we assess the recommendation performance of our framework MotifGCN.

4.5.1 Evaluating efficiency of motif adjacency matrix generation algorithms

To evaluate the efficiency of our proposed motif adjacency matrix generation algorithms, we conduct experiments using two datasets: ProgrammableWeb and Yelp. We include

Table 4.1: Comparison of motif adjacency matrix generation algorithms.

Motif	Algorithm	ProgrammableWeb		Yelp	
		Running Time	Peak Memory Usage	Running Time	Peak Memory Usage
M_2^1	VF2	0.43s	0.11 GB	/	OOM
	Ours	0.05s	0.06 GB	121.94s	7.38 GB
M_1^2	VF2	53.11s	3.24 GB	/	OOM
	Ours	8.58s	0.36 GB	345.54s	8.76 GB
M_3^1	VF2	2.33s	0.24 GB	/	OOM
	Ours	0.08s	0.06 GB	281.49s	6.96 GB
M_1^3	VF2	/	OOM	/	OOM
	Ours	8.61s	0.36 GB	346.86s	8.76 GB
M_2^{2-}	VF2	228.14s	8.36 GB	/	OOM
	Ours	33.73s	0.15 GB	10865.24s	51.90 GB
M_2^2	VF2	7.64s	0.46 GB	/	OOM
	Ours	1.9s	0.08 GB	657.56s	2.03 GB

this small dataset while ignoring the other three datasets used for model performance comparison in next section because VF2 is unable to generate any motif adjacency matrix for relatively small-sized yelp dataset due to out-of-memory (OOM). For the baseline method, we select VF2 algorithm as mentioned in Section 4.1.

Datasets. ProgrammableWeb: A mashup-API network consisting of 1,611 mashup nodes, 6,336 API nodes and 13,219 interactions. Yelp: A business review dataset containing 29,601 users, 24,734 items, and 1,517,326 interactions.

Baseline algorithm. VF2 algorithm: A well-known algorithm for subgraph isomorphism detection that is applicable to most type of graph including directed, undirected, bipartite, and homogeneous graphs.

Hardware configuration. All experiments are conducted on a computer with an AMD Ryzen 9 7900 CPU and 128GB RAM. The efficiency of evaluated methods is measured in running time and peak memory usage.

The results are shown in Table 4.1.From the comparison we can draw a conclusion that our proposed dedicated algorithms for generating motif adjacency matrices of bipartite graphs are significantly faster than the VF2 algorithm and consume less memory usage as well.

Table 4.2: Statistics of the datasets

Dataset	# of Users	# of Items	# of Interactions	Density
Yelp	29,601	24,734	1,517,326	0.00051
Gowalla	50,821	57,440	1,172,425	0.00010
Amazon	78,578	77,801	2,240,156	0.00009
Tmall	47,939	41,390	2,357,450	0.00030

4.5.2 Evaluating performance of MotifGCN

Datasets and Evaluation Metrics

The statistics of the four datasets used for performance evaluation are shown in Table 4.2. The records in each dataset are randomly split into training, validation and test sets with a ratio of 70%, 20% and 10% respectively. All models are evaluated by top-20 and top-40 recommendations with the evaluation metrics of Recall and Normalised Discounted Cumulative Gain (NDCG).

Baseline Methods

We select seven baseline methods from the perspectives of basic graph CF, higher-order-enhanced graph CF, and self-supervised graph CF methods. Although self-supervised learning is out of scope in this work, we have included this type of methods as they are commonly regarded as state-of-the-arts for CF (J. Yu, Yin et al., 2023).

- **Basic Graph CF:**
  - **NCF(X. He, Liao et al., 2017)** is an CF model leveraging MLPs to capture non-linearity in the data.
  - **LightGCN(X. He et al., 2020)** is a simplified GCN model for recommender systems. It keeps the linear propagation of node embeddings while removing all feature transformation and non-linear activation functions in the model architecture.

- **Higher-order-enhanced Graph CF:**
  - **HyperRec(J. Wang et al., 2020)** uses a series of hypergraphs from different time periods to generate dynamic node embeddings that are aware of multi-hop relationships.
  - **MHCN(J. Yu et al., 2021)** considers directed triangle motifs to incorporate multi-hop interactions and social relationships into recommendation. It also adapts attention mechanism and self-supervised learning in the training strategy.
  - **HCCF(L. Xia et al., 2022)** leverages hypergraphs to model higher-order connections and use the resulting embeddings as a view to contrast with the embeddings generated from normal graph convolution.
- **Self-supervised Graph CF:**
  - **SGL(J. Wu et al., 2021)** is a self-supervised CF model that constructs contrasting views by randomly dropout edges or nodes from the graph.
  - **SimGCL(J. Yu et al., 2022)** is a self-supervised CF model that constructs contrasting views by simply adding noise to the node embeddings.

Hyperparameter Settings

We compare our results with those reported in ("LightGCL: Simple Yet Effective Graph Contrastive Learning for Recommendation", 2022) which also used the four datasets for performance evaluation. The hyperparameters of all baseline methods are tuned according to the original papers, except that all models have the following same settings: Embeddings size is 32; the batch size is 256; the number of MLPs or graph convolution layers is two.

Table 4.3: Performance comparison with baseline methods.

Dataset	Metric	NCF	LightGCN	HyperRec	MHCN	HCCF	SGL	SimGCL	**MotifGCN**
Yelp	Recall@20	0.0252	0.0482	0.0472	0.0503	0.0626	0.0526	0.0718	**0.0777**
	NDCG@20	0.0202	0.0409	0.0395	0.0424	0.0527	0.0444	0.0615	**0.0663**
	Recall@40	0.0487	0.0803	0.0791	0.0826	0.1040	0.0869	0.1166	**0.1274**
	NDCG@40	0.0289	0.0527	0.0522	0.0544	0.0681	0.0571	0.0778	**0.0846**
Gowalla	Recall@20	0.0171	0.0985	0.0901	0.0955	0.1070	0.1030	0.1357	**0.1672**
	NDCG@20	0.0106	0.0593	0.0498	0.0574	0.0644	0.0623	0.0818	**0.0994**
	Recall@40	0.0216	0.1431	0.1356	0.1393	0.1535	0.1500	0.1956	**0.2391**
	NDCG@40	0.0118	0.0710	0.0660	0.0689	0.0767	0.0746	0.0975	**0.1183**
Amazon	Recall@20	0.0142	0.0319	0.0302	0.0296	0.0322	0.0327	0.0474	**0.0684**
	NDCG@20	0.0085	0.0236	0.0225	0.0219	0.0247	0.0249	0.0360	**0.0523**
	Recall@40	0.0223	0.0499	0.0432	0.0489	0.0525	0.0531	0.0750	**0.1071**
	NDCG@40	0.0133	0.0290	0.0246	0.0284	0.0314	0.0312	0.0451	**0.0651**
Tmall	Recall@20	0.0082	0.0225	0.0233	0.0203	0.0314	0.0268	0.0473	**0.0577**
	NDCG@20	0.0059	0.0154	0.0160	0.0139	0.0213	0.0183	0.0328	**0.0401**
	Recall@40	0.0140	0.0378	0.0350	0.0340	0.0519	0.0446	0.0766	**0.0927**
	NDCG@40	0.0079	0.0208	0.0199	0.0188	0.0284	0.0246	0.0429	**0.0522**

For MotifGCN, we search the learning rate from $\{1e^{-4}, 1e^{-3}, 1e^{-2}\}$. The L_2 regularisation term is searched from $\{1e^{-5}, 1e^{-4}, 1e^{-3}\}$. The motif type is searched from $\{M_2^1, M_1^2, M_3^1, M_1^3, M_2^{2-}, M_2^2\}$.

4.5.3 Performance Comparison

We present the performance comparison in Table 4.3. From the results, we can observe that MotifGCN shows superior performance against all baseline methods including the state-of-the-art self-supervised learning methods.

The improvement of MotifGCN over its backbone model LightGCN is significant: around 60% - 70% on Yelp and Gowalla and more than 100% on Amazon and Tmall for all evaluation metrics. This performance improvement could be attributed to network motif's high efficiency of capturing higher-order information within single layer.

Comparing to the three higher-order-enhanced graph CF baseline methods, our

framework does not rely on any attention mechanism or self-supervised learning strategy yet outperforms them by a large margin. The superiority of MotifGCN indicates that our adopted undirected network motifs can better enhance graph CF models than general hypergraph approaches(J. Wang et al., 2020; L. Xia et al., 2022).

4.6 Discussion

In this section, we discuss how motif adjacency matrices are related to powers of the original adjacency matrix and how motif adjacency matrices change the behaviour of graph convolution layer.

A motif adjacency matrix is not a trivial partition or transformation of the original adjacency matrix. Due to the definition of motif adjacency matrix, any two nodes in the motif become connected in the motif adjacency matrix regardless of the actual connectivity in the graph. This means motif adjacency matrices can enhance the message-passing process, as a node can aggregate information from distant nodes that are only reachable after multiple layers with the original adjacency matrix. In addition, the elements in the matrix are counts of motif instances where node pairs appear together, so most of the elements are different while in the original adjacency matrix elements are either one or zero. This is effectively a redistribution of the weights in neighbourhood aggregation where nodes that appear in more motif instances together with current node have larger weights.

A series of previous work (F. Wu et al., 2019; H. Zhu & Koniusz, 2020; X. He et al., 2020) simplify the traditional message-passing architecture by eliminating linear transformation, non-linear activation and self-loops from the graph convolution layers and have powers of the adjacency matrix for the neighbourhood aggregation. According to algebraic graph theory, an element of powers of adjacency matrix, $(\mathbf{A}^n)_{ij}$, is the number of walks from i to j in the graph with length n (Godsil & Royle, 2001). For

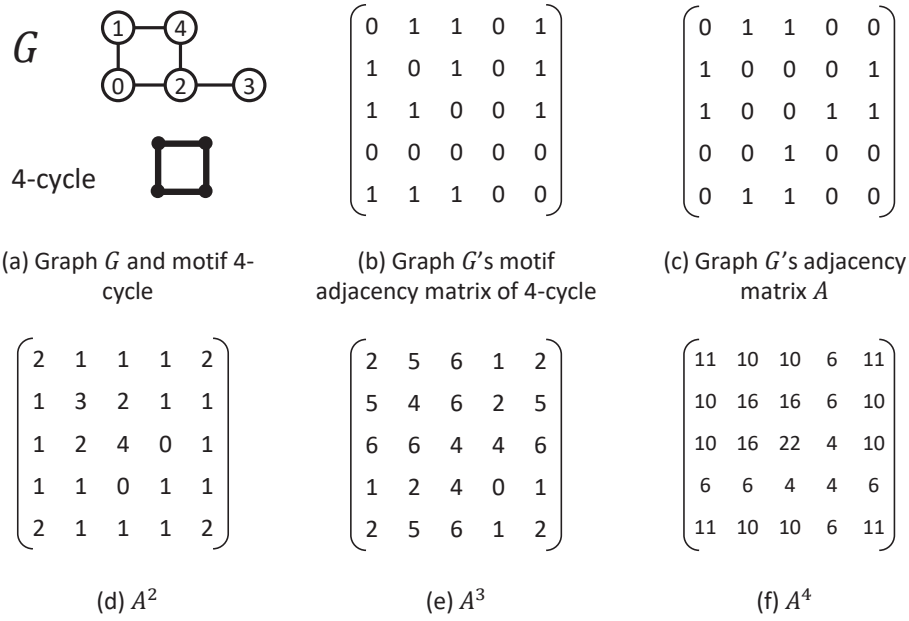


Figure 4.5: An example showing the difference between powers of adjacency matrix and motif adjacency matrix.

node i, a larger number of walks to j than to other nodes is an indication of j being structurally more important than other nodes for i. The walks include all possible pathways within the walk length so some of them can construct certain motifs. In other words, the powers of adjacency matrix contain information of various motifs. However, a large proportion of these walks contain repeated nodes and edges, which fails to reflect the distinctive impacts of different local structures to graph convolution. In addition, the number of walks is highly dependent on the parity of the walk length, causing imbalanced neighbourhood aggregation for different layers.

In comparison, the elements in a motif adjacency matrix only depend on the structure itself. Figure 4.5 illustrates an example of the difference between powers of adjacency matrix and motif adjacency matrix. Figure 4.5 (a) is simple graph where sequentially connected nodes 0, 1, 4, and 2 form an instance of a motif called four-cycle. In Figure 4.5 (b), nodes 0, 1, 2, and 4 have equal element values between each other in the motif adjacency matrix as they appear in a same motif instance.

In Figure 4.5 (c)-(f), we can see that all nodes eventually are able to aggregate information from each other and some element values grow faster than others as the exponent increases, especially node 2 who has the largest degree in the graph. This observation is consistent with the analysis of the over-squashing issue by Xu et al. (K. Xu et al., 2018) . By contrast, a motif adjacency matrix captures higher-order interactions for selective nodes depending on the motif structure, without fast increasing edge weights as in powers of adjacency matrix. This observation implies motif adjacency matrices are able to alleviate the issue of over-squashing by limiting the number of neighbours and the weights of neighbourhood information. As a result, MotifGCN is able to achieves significantly improved performance.

4.7 Conclusion

In this work, we proposed to use network motifs to reinforce the architecture of GCNs motivated by an analysis on the effect of shortcuts in GCNs. We studied the network motifs on bipartite graphs and proposed dedicated algorithms for generating motif adjacency matrices which have a significantly lower time complexity than existing approaches. We then presented MotifGCN, a novel framework for recommender systems that incorporates motif adjacency matrices in graph convolution design. We conducted experiments on four real-world datasets and showed that motifs can significantly improve a simple GCN's performance on recommendation tasks. While our method is designed primarily for bipartite graphs, the underlying principles can be easily extended to homogeneous graphs as well. Applications on homogeneous graphs and extra information such as user-user social connections are beyond the scope of this work.

Chapter 5

Motif-Guided Graph Contrastive Learning for Recommendation

5.1 Introduction

Graph-based collaborative filtering has gained considerable attention within recommender systems due to its effectiveness in modelling complex user-item interactions through graph neural networks (GNNs) (X. Wang, He, Wang et al., 2019; X. He et al., 2020; W. Yu et al., 2022). The core strength of graph-based collaborative filtering lies in its capacity to represent users and items as interconnected nodes, enabling efficient exploitation of relational data and structural dependencies. This approach has notably improved predictive accuracy and facilitated richer interaction modelling compared to traditional recommendation methods.

Recent advancements (J. Wu et al., 2021; K. Zhou et al., 2020) in recommender systems have introduced Graph Contrastive Learning (GCL), a self-supervised framework designed to enhance robustness and representation quality. GCL operates by generating multiple augmented graph views through perturbation strategies and maximising agreement between embeddings derived from these diverse views. This process has

shown significant potential in addressing prominent challenges such as data sparsity, susceptibility to noisy interactions, and incomplete information. Consequently, GCL techniques have shown improved generalisation and robustness in various recommendation scenarios (Z. Lin et al., 2022; J. Yu et al., 2022).

However, conventional GCL methods typically employ simple and uniformly random augmentation techniques, such as node dropout or edge perturbations, which can inadvertently distort critical structural information and remove meaningful interactions. Such random augmentations have been shown to limit the quality and interpretability of learnt graph representations (H. Liang et al., 2023; Trivedi et al., 2022; C. Wei et al., 2023; Y. Liu, Kertkeidkachorn, Miyazaki & Ichise, 2025; C. Zhao et al., 2025). In response to these limitations, recent approaches have started exploring more structured and semantically meaningful augmentation methods, aiming to preserve valuable relational and structural context. For example, some methods integrate socially-informed augmentations to retain homophily signals which is crucial for social recommendations (Y. Zhang et al., 2024; W. Jiang et al., 2024), while others leverage hierarchical clustering techniques to maintain multi-resolution structural semantics (J. Xue et al., 2025; H. Wei, Wang, Ji, Guang & Yan, 2025). Additionally, explicit neighbour-aware methods have been proposed to reduce the randomness inherent in conventional sampling by considering structural neighbours as positive pairs (Z. Lin et al., 2022; J. Xu et al., 2025). Some other strategies formalise a variety of graph operations through principled message-passing frameworks or global statistical transformations to retain important local and global interaction patterns (Zhuo et al., 2024; Ni et al., 2024).

Despite the progress these structured methods represent, they commonly rely on implicit assumptions or heuristic augmentation strategies, limiting explicit interpretability and failing to differentiate effectively between strong and weak user-item interactions. To bridge this gap, we propose a novel contrastive learning framework called Motif-Guided Graph Contrastive Learning (MGGCL). MGGCL integrates the structural

semantics into graph-based collaborative filtering by constructing contrastive views with motifs, which are frequently recurrent subgraph patterns that capture multi-hop local interactions. To be specific, we select two star motifs and a rectangle motif that represent different structural semantics for recommendation graphs. After that, we prove that the rectangle-motif-induced adjacency structure is contained within the union of the structures captured by the two star motifs. This provides justification for selecting the three motifs to construct contrastive views, as our analysis demonstrates that similarities between these views capture robust correlations among users and items, while their differences emphasise sparse or isolated user preferences. We then construct contrastive views by augmenting the adjacency matrix with motif adjacency matrices so that the view embeddings are generated with motif-guided neighbourhood aggregation.

By strategically constructing and contrasting motif-specific views, MGGCL emphasises meaningful structural signals and suppresses noise from isolated or sparse interactions, substantially improving the robustness, interpretability, and performance of recommendations. Thus, MGGCL systematically addresses the interpretive shortcomings of existing random and heuristic approaches by explicitly incorporating higher-order structural semantics into contrastive view construction.

In many real-world recommendation graphs, interactions are heavily skewed. Popular users and items lead to dense star-shaped motifs that can dominate learnt representations while semantically richer co-interaction rectangle motifs remain under-represented. Our motif-guided contrastive learning framework explicitly constructs two graph views: one preserves all star motifs and another retains rectangle motifs. Given that only rectangle-engaged edges appear as positives in both views, gradient signals automatically weaken hub-only links and amplify structurally meaningful co-interaction patterns. Empirical results on four benchmarks confirm this mechanism in practice. MGGCL yields the largest gains on highly skewed Douban-Book and iFashion datasets, demonstrating significant improvements in Recall and NDCG over state-of-the-art baselines.

Our main contributions are as follows.

- We design a novel motif-guided graph contrastive learning framework that explicitly integrates structural semantics into graph view construction through motif-based augmentations. Unlike conventional stochastic perturbations, our motif-based augmentation strategy provides clear structural transparency, enabling potential interpretability and attribution of model behaviour to specific local graph patterns.

- We formally establish a hierarchical relationship between the motif adjacency matrices constructed from star motifs and rectangle motifs, where edges forming rectangles represent higher-order, structurally robust user-item interactions. By explicitly contrasting the rectangle-motif view against the star-motifs view, the model is optimised to strengthen node representations derived from these rectangle patterns where nodes have stronger correlation. Concurrently, the model naturally reduces the influence of weaker interactions that appear only in the star-motifs view, leading to improved robustness of the learnt representations.

- We demonstrate that incorporating motif-aware adjacency matrices into contrastive views enhances neighbourhood aggregation efficiency by explicitly capturing meaningful multi-hop structural relationships. This structural richness allows the model to achieve improved node embeddings through contrastive objectives, leading to better discriminative power and representation robustness compared to standard stochastic augmentation strategies.

- We conduct extensive experiments on four real-world recommendation benchmarks and achieve consistent accuracy improvements. In particular, skewed datasets benefit effectively from MGGCL's ability to reduce popularity bias from overly popular nodes and to amplify under-represented structurally meaningful

motifs. Systematic parameter sensitivity analysis further demonstrates stable performance without exhaustive hyperparameter tuning.

The rest of this chapter is organised as follows. Section 5.3 provides the background knowledge to understand our proposed method. Section 5.4 presents our proposed MGGCL framework. Section 5.5 shows our experimental results and performance analysis. Finally, Section 5.6 concludes this chapter.

5.2 Related Work

This section discusses three key areas relevant to our study: (1) GCL recommendation models with simple random augmentations; (2) GCL recommendation models that generate contrastive view using structural and semantic augmentations; and (3) the use of motifs in recommender systems.

5.2.1 GCL Recommendation Models with Simple Random Augmentations

Graph Contrastive Learning has emerged as a powerful self-supervised learning paradigm for graph-based recommendation. It focuses on learning robust user and item representations by contrasting multiple views of the interaction graph. This approach addresses data sparsity and improves generalisation without relying on explicit supervision. The foundational work by Wu et al. (J. Wu et al., 2021) introduced SGL, which creates contrastive views through random node and edge dropout. SGL encourages agreement between embeddings derived from these views, thereby mitigating sparsity and enhancing robustness. However, this stochastic augmentation strategy can distort meaningful graph structures, which leads to inconsistencies in semantics and reduced interpretability.

To address these limitations, later models explored simplified or more efficient augmentation strategies. SimGCL (J. Yu et al., 2022) avoids explicit structural perturbations by injecting Gaussian noise directly into node embeddings, preserving graph connectivity while maintaining competitive performance. SelfCF (X. Zhou et al., 2023) shifts the focus to augmentation in the output space by applying contrastive learning directly to the final embeddings. By avoiding alteration on the input graph and reliance on negative sampling, this method offers a more lightweight and efficient training process.

Several models aim to refine the contrastive signal or improve its robustness. BDCL (P. Yu et al., 2024) r applies directional perturbations, preserving sparse behavioural data without random corruption. RecDCL (D. Zhang et al., 2024) introduces both batch-wise and feature-wise contrastive objectives to improve embedding orthogonality. DCCR (R. Zhao et al., 2024) uses a self-adaptive noise mechanism and constructs dual collaborative views to extract high-value samples and enhance recommendation diversity. GGDHSCL (X. Liu et al., 2025) introduces a generative diffusion mechanism coupled with hard negative sampling, which strengthens the contrastive signal rather than modifies view structures.

Recent work has also emphasised improved handling of noisy or uninformative views. SymmetricGCL (C. Zhao et al., 2025) introduces a denoising strategy using symmetric contrastive pairs to mitigate degradation from noisy augmentations. WeightedGCL (Z. Chen et al., 2025) enhances SGL with dynamic view-level weighting that allows more informative contrastive views to contribute more significantly to learning.

To explore more stable and informative view construction, NLGCL (J. Xu et al., 2025) generates contrastive views from naturally existing neighbour layers instead of random perturbations, reducing semantic drift while maintaining graph topology. DimCL (C. Zhang et al., 2025) augments representations in a dimension-aware fashion

to encourage diverse embedding components for contrastive learning.

To support privacy and personalisation, Personalised Federated Contrastive Learning (S. Wang et al., 2025) adapts GCL to the federated setting by aggregating user-device interactions without centralising user data. This design introduces personalisation and scalability but requires careful calibration to maintain embedding consistency across clients.

Collectively, these works illustrate a trajectory from random augmentations toward more principled and robust contrastive learning techniques. Yet, many still operate with implicit structure and offer limited topological semantics, motivating structurally grounded alternatives.

5.2.2 Methods with Structured and Semantically Guided Augmentations

Recognising the limitations of uniform random perturbations, several researchers have explored structured or semantically informed augmentations. SG-CLR (Y. Zhang et al., 2024) enhances the semantic integrity of contrastive views by guiding edge additions based on homophily. HGCL (J. Xue et al., 2025) takes a hierarchical perspective by clustering items into multiple resolution layers to preserve coarse-to-fine structural semantics. NCL (Z. Lin et al., 2022) advances the idea of local structure by treating topological neighbours as natural positives rather than sampling corrupted ones. This strategy captures contextual consistency and reduces semantic drift during augmentation.

JCG (Dai et al., 2024) combines structural and semantic contrastive objectives to address data imbalance and reduce the influence of high-degree nodes. This improves generalisation across heterogeneous graph distributions. MD-GCCF (X. Li et al., 2024) advances multi-view learning by integrating deep collaborative signals and

designing both local and global contrastive objectives. By mitigating over-smoothing and reducing redundancy, this architecture yields more informative representations for recommendation.

Several models introduce learnable or deterministic transformation schemes. CL-KDM (Ni et al., 2024) bypasses traditional dropout strategies entirely by maximising statistical dependence between views, which preserves global latent geometries without edge removal. LMACL (X. Liu et al., 2024) combines GCN and GAT to learn augmentation strategies to generate multi-hop and neighbour-aware contrastive views. These methods present promising steps toward augmentations that encode high-quality semantics or structure. However, many rely on heuristics or predefined rules and rarely integrate clear higher-order graph patterns such as motifs into the view generation process.

5.2.3 Motif-Based Approaches for Graph Recommendation and Contrastive Learning

Motifs are recurrent subgraph structures such as stars and rectangles that offer a principled way to capture higher-order relationships and encode meaningful local semantics in graphs. Although extensively studied in biological and social networks, motifs have only recently been adopted in recommender systems. Their ability to reflect structured behavioural patterns, such as co-engagement, item co-consumption, or user hubs, makes them suitable for addressing sparsity and noise.

Recent studies have explored motif-based enhancements in recommender systems. In Chapter 3, we extend Graph Attention Networks by integrating motif structures to capture high-order service relationships. In Chapter 4, we proposes a framework that employs star and rectangle motifs to define richer neighbourhoods in collaborative filtering. Wang et al. Motifs-Res (Y. Sun et al., 2022) formulates motif co-occurrence

as hyperedges and applies hypergraph convolution with contrastive loss to exploit local motif semantics. MotifSR (Cui et al., 2021) incorporates motif-enhanced temporal subgraphs to better capture session-level user behaviours in sequential recommendation. These models demonstrate the growing interest in exploiting motif semantics to strengthen signal quality and model generalisation in graph-based recommendation.

Complementing this, motif-based contrastive learning has been studied in other domains. Motif-driven contrastive learning of graph representations (S. Zhang et al., 2024) introduces motif-induced views to enforce global and local consistency in general graph representation learning, where motif co-occurrence defines semantically consistent neighbourhoods. MotifCC (X. Wu et al., 2024) proposes a framework that builds subgraphs from triangle motifs and applies contrastive learning between lower-order and higher-order views to jointly preserve structural uniqueness and encourage clustering separability. MHGCL (C. Li et al., 2025) jointly leverages motif structures and node homogeneity to refine the selection of contrastive pairs, thereby suppressing noise and improving representation discrimination in general graph tasks. Although effective, these works do not target recommender systems and typically operate in node classification or community detection settings.

Our work departs from the above by being the first to integrate motif-based structures into contrastive view generation specifically for collaborative filtering. Instead of treating motifs as auxiliary enhancements or post-processing filters, we construct motif-induced adjacency matrices and use them as structurally meaningful and interpretable views. Rectangle motifs capture higher-order co-engagement signals, while the combination of star motifs additionally includes sparse preferences or transient behaviours. By contrasting these motif-guided views, our approach prioritises robust local structures, reduces potential noise arisen from sparse or isolated patterns, and facilitates transparent recommendation. This approach uniquely unifies interpretability, semantic augmentation, and robustness within the contrastive learning framework for

graph-based recommendation.

5.3 Preliminaries

5.3.1 Motifs in Recommendation Graphs

In recommender system research, user-item interactions are typically represented using bipartite graphs. Bipartite graphs naturally fit collaborative filtering tasks since they capture the relationships between two distinct node sets, i.e., users and items. The formal definition of bipartite graphs is provided below.

Definition 4 (Bipartite Graphs (Diestel, 2017))**.** *A bipartite graph $G = (U, V, E)$ is a graph where all nodes in G are partitioned into two disjoint classes U and V, i.e. $U \cap V = \emptyset$, such that every edge has its terminal node in different classes from its initial node: nodes in the same partition class must not be adjacent.*

Motifs are significant patterns that frequently recur within graphs, appearing more frequently than expected in randomised graphs (Milo et al., 2002). Since motifs typically include more than two nodes, they offer valuable insight into the local structural semantics of the graph while simultaneously influencing its global characteristics. Figure 1 illustrates bipartite motifs composed of up to four nodes. Users are represented by circles, and items by squares. Following the naming system proposed in the previous chapter, these motifs are named M_b^a where a indicates the number of user nodes in the motif while b indicates the number of item nodes in the motif. For ease of understanding, we refer to these motifs according to their topological structures: M_2^1, M_3^1, M_1^2, and M_1^3 are termed star motifs, M_2^{2-} is termed a path motif, and M_2^2 is termed a rectangle motif.

A common approach to incorporate network motifs into graph learning frameworks is to construct motif adjacency matrices.

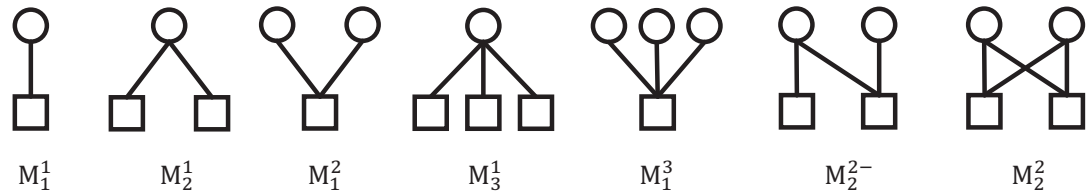


Figure 5.1: All motifs consisting of two to four nodes in bipartite graphs. The superscript indicates the number of user nodes in the motif, whereas the subscript denotes the number of item nodes. The hyphen in M_2^{2-} indicates one less edge in the motif compared to M_2^2.

Definition 5 (Motif Adjacency Matrix (Benson et al., 2016))**.** *Given a graph* $G = (V, E)$ *and a motif* m*, the motif adjacency matrix* $A_m \in \mathbb{N}^{|V|\times|V|}$ *is defined such that* $(A_m)_{ij}$ *equals the number of instances of motif* m *in which nodes* i *and* j *co-occur.*

5.3.2 Contrastive Learning for Recommender Systems

In the context of graph-based collaborative filtering, Contrastive Learning (CL) operates by generating multiple views of the user-item interaction graph through augmentations such as edge or node dropout. These augmented views are processed by graph encoders to produce representations, and the model is trained to maximise the agreement between representations of the same node across different views while distinguishing between different nodes.

The joint learning scheme used in CL methods is a combination of the recommendation loss $\mathcal{L}_{\text{rec}}$ and the contrastive loss $\mathcal{L}_{\text{cl}}$:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \lambda\mathcal{L}_{\text{cl}}, \tag{5.1}$$

where the CL task serves as an auxiliary objective, with its influence controlled by a hyperparameter λ. For the recommendation loss, a common choice is the Bayesian

Personalised Rank (BPR) loss:

$$\mathcal{L}_{\text{rec}} = -\sum_{u,i\in\mathcal{B}} \log(\sigma(\mathbf{z}_u^\top \mathbf{z}_i - \mathbf{z}_u^\top \mathbf{z}_j)), \tag{5.2}$$

where $\mathcal{B}$ is the sampled batch, σ is the sigmoid function, $\mathbf{z}_u$ is the embedding vector for user u, $\mathbf{z}_i$ and $\mathbf{z}_j$ are embedding vectors for positive item i and negative item j respectively.

The common choice for contrastive loss is the InfoNCE loss:

$$\mathcal{L}_{\text{cl}} = -\sum_{i\in\mathcal{B}} \log \frac{\exp\left(\mathbf{z}_i'^\top \mathbf{z}_i''/\tau\right)}{\sum_{j\in\mathcal{B}} \exp\left(\mathbf{z}_i'^\top \mathbf{z}_j''/\tau\right)}, \tag{5.3}$$

where $\mathbf{z}_i'$ and $\mathbf{z}_i''$ are two different representations of the same sample i typically obtained by applying different augmentations or views, $\mathbf{z}_j''$ is the second representation of sample j in the batch, τ is the temperature parameter that controls the sharpness of the distribution.

5.4 Methodology

In this section, we present our MGGCL framework. We first explain our motivation of using motifs in contrastive learning to learn structural semantics, and then introduce the design of model architecture.

5.4.1 Motif-Guided Structural Semantics

Traditional contrastive learning in recommendation systems often relies on random edge dropout or uniform augmentation to generate views, which risks discarding meaningful interaction patterns or amplifying noise. Our approach instead leverages motif-guided structural semantics to create semantically distinct views. Specifically, we choose the three motifs with different structural semantics: M_2^1 implies potential similarity or

complementarity between items, M_1^2 reflects possible shared interests between users, and M_2^2 indicates stronger relevance among users and items. We do not choose M_3^1, M_1^3 and M_2^{2-} for the following reasons: M_3^1 and M_1^3 are expanded versions of M_2^1 and M_1^2 respectively, sharing similar structural semantics but excluding isolated or peripheral sparse subgraphs present in M_2^1 and M_1^2; although M_2^{2-} contains structures of M_2^1 and M_1^2, it blurs individual-level signals by mixing two users via a high-degree item or two items by a "shopaholic" user.

By contrasting views constructed with M_2^2 and the combination of M_2^1 and M_1^2, we explicitly decouple two fundamental aspects of user-item interactions: (1) star motifs, i.e. M_2^1 and M_1^2, which reflect preferences or transient behaviours, and (2) rectangle motif, i.e. M_2^2, which encodes robust group behaviours such as item communities or shared user tastes. The contrast arises from the observation that edges appearing in M_2^1 and M_1^2 but absent in M_2^2 often correspond to low-consistency signals, as their interactions are not strengthened by broader user–item coalitions. These differences are inherently meaningful as they capture less common behaviours or noises that lack reinforcement from similar user-item interactions. By contrasting these views, the model learns to prioritise interactions validated by both local and global structural contexts while retaining sensitivity to sparse but informative signals. This approach avoids the interpretability pitfalls of empirical random augmentation, as the views directly align with observable interaction semantics in real-world bipartite systems.

5.4.2 Structural Hierarchy between Motif Adjacency Matrices

We briefly formalise the structural hierarchy between motif adjacency matrices used to define the two contrastive views:

Proposition 2. *Let A_{12} and A_{21} denote the motif adjacency matrices derived from star motifs M_2^1 and M_1^2 respectively, and let A_{22} denote the motif adjacency matrix*

corresponding to the rectangle motif M_2^2. Then:

$$supp(A_{22}) \subseteq supp(A_{12} \cup A_{21})$$

where $supp(A)$ is the set of index pairs (i, j) for which $A_{ij} \neq 0$.

Proof. Consider a rectangle motif instance with users u_1, u_2 and item i_1, i_2. It contains the four edges $(u_1, i_1), (u_1, i_2), (u_2, i_1), (u_2, i_2)$. Each of these edges belongs to at least one of M_2^1 or M_1^2. Edge (u_1, i_1) appears in an M_2^1 instance (u_1, i_1, i_2) and an M_1^2 instance (i_1, u_1, u_2). Analogous reasoning holds for the other three edges.

If $(p, q) \in supp(A_{22})$, then some instances of A_2^2 contain (p, q), and the above guarantees that (p, q) appears in at least one instance of M_2^1 or M_1^2. Hence $(p, q) \in supp(A_{12})$ or $(p, q) \in supp(A_{21})$, which implies:

$$(p, q) \in supp(A_{12} \cup A_{21})$$

Since every edge in $supp(A_{22})$ is also in $supp(A_{12} \cup A_{21})$, then:

$$supp(A_{22}) \subset supp(A_{12} \cup A_{21})$$

When every instance of M_2^1 and M_1^2 is "closed" by at least one instance of M_2^2, $supp(A_{22}) = supp(A_{12} \cup A_{21})$.

Hence we have:

$$supp(A_{22}) \subseteq supp(A_{12} \cup A_{21})$$

□

Remark 1. *The inclusion in Proposition 1 is* strict *for the majority of all real-world graphs. Equality can occur only in the degenerate case where* every *edge that participates in an instance of a M_2^1 or M_1^2 also belongs to at least one instance of M_2^2.*

Equivalently, every two-hop user-item path ($u \to i \to u$ or $i \to u \to i$) is "closed" by an additional user or item, forming a M_2^2. Such a closure is highly unlikely in the sparse interaction graphs typical of recommender systems, so we treat the strict inclusion case $supp(A_{22}) \subset supp(A_{12} \cup A_{21})$ as the practical default.

By definition, any two users connected to two common items inherently contain the edge connections constituting star motifs as substructures. Specifically, the rectangle motif can always be decomposed into combinations of star motifs, guaranteeing hierarchical inclusion. Consequently, this structural nesting explicitly isolates the incremental higher-order interactions captured by rectangles, clearly delineating which local structural signals each contrastive view encodes. This deterministic hierarchical structure significantly narrows the potential explanation space. Although direct interpretability experiments are beyond the current scope, the transparent construction mechanism provides a clear path for future analyses, such as motif attribution and saliency enrichment, which can systematically reveal the influence of these motifs.

5.4.3 Motif-Guided Graph Contrastive Learning

In this section, we present the architecture of our framework Motif-Guided Graph Contrastive Learning (MGGCL), as illustrated in Figure 5.2. Our approach leverages motifs to construct contrasting views for self-supervised learning in recommendation graphs. Let $\mathcal{G} = (\mathcal{U}, \mathcal{V}, \mathcal{E})$ denote a bipartite graph with users $\mathcal{U}$, items $\mathcal{V}$, and edges $\mathcal{E}$ representing interactions. We select the following three motifs to capture distinct interaction structures:

- **Motif** M_2^1: User-centric co-interactions, represented by pairs $\{(u, i_1), (u, i_2)\}$ where a user u interacts with two items i_1, i_2.
- **Motif** M_1^2: Item-centric co-interactions, represented by pairs $\{(u_1, i), (u_2, i)\}$ where two users u_1, u_2 interact with the same item i.

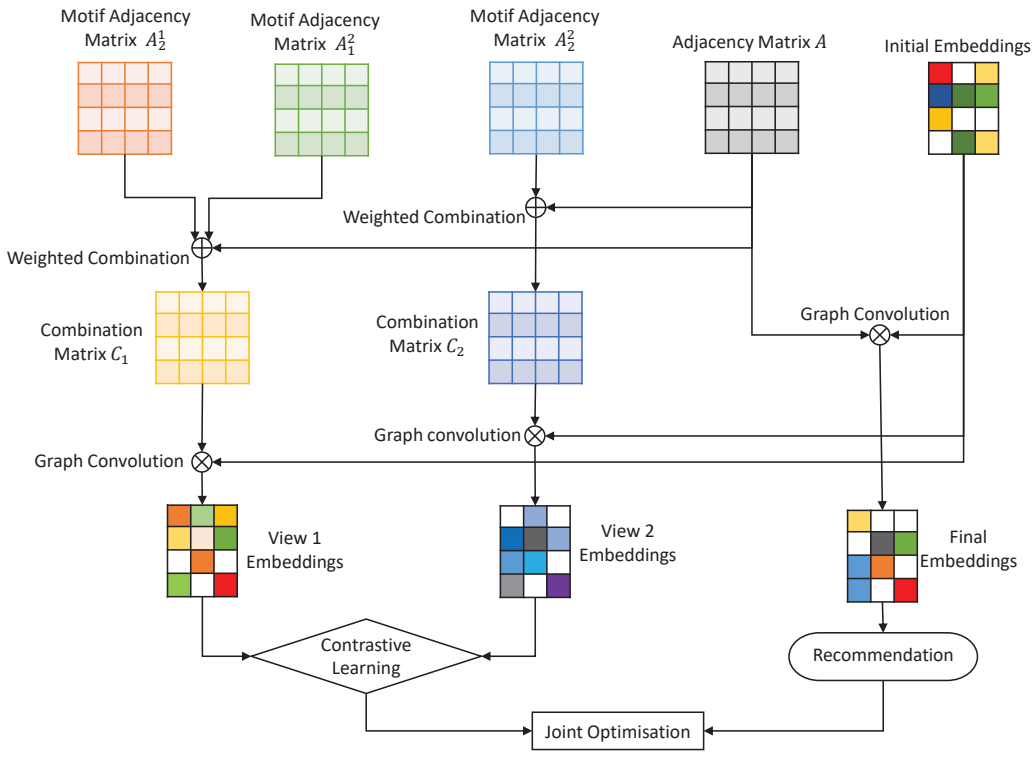


Figure 5.2: The architecture of Motif-Guided Graph Contrastive learning framework.

- **Motif** M_2^2: Dense bipartite hubs, represented by a 2×2 complete bipartite graph $\{(u_1, i_1), (u_1, i_2), (u_2, i_1), (u_2, i_2)\}$. These motifs are stronger collaborative signals since they reflect group behaviour or shared preferences.

Two contrastive views are generated by combining motif adjacency matrices with the original adjacency matrix $\mathbf{A}$:

$$\mathbf{C}_1 = (1-\theta) \cdot \widetilde{\mathbf{A}} + \theta \cdot \frac{\widetilde{\mathbf{A}}_2^1 + \widetilde{\mathbf{A}}_1^2}{2}, \tag{5.4}$$

$$\mathbf{C}_2 = (1-\theta) \cdot \widetilde{\mathbf{A}} + \theta \cdot \widetilde{\mathbf{A}}_2^2, \tag{5.5}$$

where $\widetilde{\mathbf{A}}$ is normalised adjacency matrix; $\widetilde{\mathbf{A}}_2^1$, $\widetilde{\mathbf{A}}_1^2$, and $\widetilde{\mathbf{A}}_2^2$ are normalised motif adjacency matrices of the three corresponding motifs; $\theta \in (0, 1]$ is the motif ratio that balances the contribution between the original adjacency matrix and the motif adjacency matrices, which enables the model to adjust to graphs with varying levels of motif density. $\mathbf{C}_1$ emphasises sparse pairwise interactions, while $\mathbf{C}_2$ amplifies dense hub structures. Both matrices are row-normalised to stabilise training.

Node embeddings are generated using a shared graph encoder $f_\theta(\cdot)$ based on LightGCN (X. He et al., 2020), which avoids nonlinearities and normalisation for efficiency:

$$\mathbf{H}^{(l)} = \mathbf{C} \cdot \mathbf{H}^{(l-1)}, \tag{5.6}$$

where $\mathbf{C} \in \{\mathbf{C}_1, \mathbf{C}_2\}$ and $\mathbf{H}^{(l)}$ is the l-th layer embedding. The final embedding $\mathbf{z}_u$ for user u (and analogously for items) is obtained by averaging all layer outputs:

$$\mathbf{z}_u = \frac{1}{L} \sum_{l=1}^{L} \mathbf{H}_u^{(l)}. \tag{5.7}$$

By augmenting the adjacency matrix with motif-induced adjacency matrices, the graph convolution layer is guided by higher-order structural patterns and achieves more

efficient neighbourhood aggregation. This allows the contrastive views to extend their receptive field without increasing the number of graph convolution layers. As a result, the contrastive learning process becomes more reflective of both local and global graph structures, thereby enhancing the overall performance of the framework.

We employ a noise-tolerant contrastive loss to maximise mutual information between user and item embeddings across views while suppressing motif discrepancies. For a user u, embeddings $\mathbf{z}_u^{C_1}$ and $\mathbf{z}_u^{C_2}$ from $\mathbf{C}_1$ and $\mathbf{C}_2$ form a positive pair. Negative pairs are sampled from other users/items in the batch. The loss is defined as:

$$\mathcal{L}_{\text{cl}} = -\sum_{u \in \mathcal{U}} \log \frac{\exp\left(\phi(\mathbf{z}_u^{C_1}, \mathbf{z}_u^{C_2})/\tau\right)}{\sum_{v \in \mathcal{U}} \exp\left(\phi(\mathbf{z}_u^{C_1}, \mathbf{z}_v^{C_2})/\tau\right)}, \tag{5.8}$$

where $\phi(\cdot)$ measures cosine similarity and τ is a temperature hyperparameter. The full objective combines $\mathcal{L}_{\text{ssl}}$ with a BPR loss $\mathcal{L}_{\text{rec}}$:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \lambda \mathcal{L}_{\text{cl}}, \tag{5.9}$$

where λ controls the contribution of contrastive learning signals. The model retains all edges but implicitly reduces the weights of interactions exclusive to M_2^1 and M_1^2, treating them as structured noise relative to the dominant hub structures in M_2^2.

5.5 Experiments

5.5.1 Experimental Settings

We conduct experiments on four public benchmark datasets: Douban-Book (J. Yu et al., 2022), iFashion (J. Wu et al., 2021), Amazon-Kindle (J. Yu, Xia et al., 2023) and Amazon-Book (X. He et al., 2020). We follow prior work in using the Gini coefficient to measure inequality in item distribution (Abdollahpouri, Mansoury, Burke, Mobasher

Table 5.1: Statistics of the datasets

Dataset	# of Users	# of Items	# of Edges	Density	User-Gini	Item-Gini
Douban-Book	12859	22294	598420	0.00209	0.624	0.6455
iFashion	300000	81614	1607813	0.00007	0.3569	0.7666
Amazon-Kindle	138333	77801	1909965	0.00014	0.5426	0.5422
Amazon-Book	52643	91599	2984108	0.00062	0.4468	0.4617

& Malthouse, 2021) and extend it to quantify both user and item popularity inequality in our datasets. The statistics of these datasets are presented in Table 5.1. The records in each datasets are randomly split into training, validation and testing set with a ration of 7:1:2. All models are evaluated using two common metrics: Recall@K and NDCG@K with K=5, 10, 20, 40. All results are based on the average of five experimental trials.

5.5.2 Baselines

We select the following baseline models for performance comparison.

- **LightGCN (X. He et al., 2020)** is a simplified GCN model for recommender systems. It keeps the linear propagation of node embeddings while removing all feature transformation and non-linear activation functions in the model architecture.
- **NCL (Z. Lin et al., 2022)** is a model that boosts graph collaborative filtering by using contrastive learning enriched with neighbourhood information.
- **SGL (J. Wu et al., 2021)** is a self-supervised CF model that constructs contrasting views by randomly dropout edges or nodes from the graph.
- **SelfCF (X. Zhou et al., 2023)** is a self-supervised framework that improves collaborative filtering by augmenting output embeddings without negative sampling.

Table 5.2: Performance comparison.

Dataset	Model	Recall@5	NDCG@5	Recall@10	NDCG@10	Recall@20	NDCG@20	Recall@40	NDCG@40
Douban-Book	LightGCN	0.05131	0.09851	0.08345	0.09868	0.12720	0.10662	0.18881	0.12462
	NCL	0.06189	0.10724	0.09416	0.10696	0.13840	0.11534	0.19833	0.13326
	SGL	0.02765	0.04562	0.04267	0.04653	0.06066	0.04962	0.08224	0.05543
	SelfCF	0.03269	0.06171	0.05577	0.06306	0.08698	0.06954	0.13724	0.08474
	SimGCL	0.04641	0.07628	0.06822	0.07634	0.09862	0.08245	0.13711	0.09403
	MGGCL	**0.07910**	**0.14342**	**0.11548**	**0.13882**	**0.16124**	**0.14424**	**0.21507**	**0.15844**
iFashion	LightGCN	0.03396	0.02338	0.05338	0.02985	0.08137	0.03726	0.11999	0.04557
	NCL	0.02428	0.01685	0.03831	0.02153	0.05923	0.02707	0.08953	0.03358
	SGL	0.03956	0.02763	0.06028	0.03455	0.08959	0.04230	0.12953	0.05090
	SelfCF	0.02489	0.01695	0.04006	0.02200	0.06362	0.02819	0.09945	0.03585
	SimGCL	0.03748	0.02644	0.05628	0.03273	0.08175	0.03948	0.11541	0.04676
	MGGCL	**0.04929**	**0.03455**	**0.07479**	**0.04307**	**0.10945**	**0.05224**	**0.15567**	**0.06221**
Amazon-Kindle	LightGCN	0.07224	0.06478	0.10752	0.07673	0.15187	0.09029	0.20721	0.10493
	NCL	0.05060	0.04338	0.08057	0.05382	0.12128	0.06624	0.17040	0.07934
	SGL	0.06228	0.05520	0.09737	0.06728	0.14247	0.08097	0.19977	0.09612
	SelfCF	0.01919	0.01630	0.03084	0.02040	0.05053	0.02637	0.07936	0.03383
	SimGCL	0.06732	0.05973	0.10175	0.07142	0.14468	0.08445	0.19721	0.09834
	MGGCL	**0.07250**	**0.06572**	**0.10900**	**0.07793**	**0.15591**	**0.09209**	**0.21230**	**0.10701**
Amazon-Book	LightGCN	0.01096	0.01884	0.01955	0.02096	0.03407	0.02664	0.05754	0.03548
	NCL	0.01115	0.01911	0.02015	0.02158	0.03506	0.02743	0.05871	0.03636
	SGL	0.01252	0.02133	0.02226	0.02384	0.03779	0.02985	0.06284	0.03925
	SelfCF	0.00724	0.01344	0.01305	0.01465	0.02228	0.01809	0.03718	0.02365
	SimGCL	0.01404	0.02352	0.02436	0.02593	0.04062	0.03223	0.06454	0.04125
	MGGCL	**0.01458**	**0.02502**	**0.02570**	**0.02767**	**0.04301**	**0.03431**	**0.06981**	**0.04439**

- **SimGCL (J. Yu et al., 2022)** is a self-supervised CF model that constructs contrasting views by simply adding noise to the node embeddings.

5.5.3 Hyperparameter Settings

The hyperparameters of all baseline methods are tuned according to the original papers, except that all models have the following same settings: Embeddings size is 32; the batch size is 4096; the number of MLPs or graph convolution layers is two.

For MGGCL, we search for the learning rate from $\{1e^{-4}, 1e^{-3}, 1e^{-2}\}$. The L_2 regularisation term is searched from $\{1e^{-5}, 1e^{-4}, 1e^{-3}\}$.

5.5.4 Performance Comparison

We present the performance comparison in Table 5.2. Our proposed MGGCL specifically leverages motif-guided structural semantics, distinguishing it from traditional contrastive learning methods that rely on random edge dropout or uniform augmentation. Evaluating the results across four real-world datasets, MGGCL demonstrates consistent superiority compared to baseline methods.

Douban-Book:

Among the four benchmark datasets, Douban-Book exhibits the highest interaction density, with users engaging in approximately 46.5 rating events on average and each item receiving 26.8 ratings. Nonetheless, item-popularity Gini coefficient of around 0.65 indicates a long-tail skewed distribution. This indicates that some popular books receive disproportionately high attention compared to the majority of other niche items. The user-side distribution is of a similar nature as the user-popularity Gini coefficient is 0.62, which shows that a small proportion of power users contribute a particularly large number of ratings. The inequalities for user and item sides show that interaction patterns are not uniformly distributed. Instead, hubs and clusters are displayed.

Considering the moderate connectivity and long-tailed popularity displayed in the dataset, star shaped motifs around bestsellers and rectangle motifs among reader niches are both well-represented in the interaction graph. Hence, MGGCL can use the motif abundance in the dataset to refine neighbourhood aggregation, which then effectively leads to enhanced recommendation performance. As shown in Table 2, the Recall@5 and NDCG@5 are 27.8% and 33.7% higher than the strongest baseline NCL. The model's explicit rectangle-vs-star contrast particularly boosts NDCG. This shows that even in datasets with already abundant user-item interactions, the model achieves additional performance gains by focusing on higher-order and structurally closed patterns.

iFashion:

iFashion is the sparcest dataset in our study. It combines a significantly large user base of 300,000 with only around 1.6 million interactions. With a user popularity Gini coefficient of 0.36, the user activity is relatively evenly spread. In contrast, a high item popularity Gini coefficient of 0.77 reveals extreme popularity skewness. Only a small fraction of fashion items dominate the total amount of purchases, while the vast majority of items see very little demand. Traditional methods struggle to separate signals from noise in such sparce interactions, whereas MGGCL's motif-guided view construction manages to surface local dense clusters, e.g. users co-purchasing the same style of products. Consequently, MGGCL outperforms the strongest baseline by 22.2% on Recall@40, which confirms its ability to amplify rare yet semantically rich co-interaction patterns in ultra-sparse settings.

Amazon-Kindle:

Amazon-Kindle has a second lowest density in the four datasets, with moderate user and item popularity Gini coefficients around 0.54. This reflects a balance between devoted e-book enthusiasts and the broad distribution of interest across genres and series. In terms of experiment results, MGGCL consistently outperforms all GCL baselines but is only marginally better than LightGCN. A possible reason is that when popularity skewness is moderate, the raw first-order adjacency used by LightGCN captures a large share of the meaningful signal without being strongly affected by hub bias.

Amazon-Book:

The Amazon-Book dataset is consistent of a large item catalogue of around 92,000 but fewer users, which produces a moderate sparsity level. Among the four datasets, the sparsity level of Amazon-Book is in between Douban-Book and Amazon-Kindle. With relatively low Gini coefficients for both user and item popularities, the dataset shows the most uniform interaction patterns of the four. Despite this, previous research (Shi et al., 2024; W. Xu, Gao, Su, Wu & Wang, 2024) show that the dataset still shows considerable

popularity bias. As a result, MGGCL shows less improvements than Douban-Book and iFashion but still outperform the strongest baseline by 8% higher on Recall@40 and 7.6% higher on NDCG@40.

Summary:

Across datasets spanning two orders of magnitude in density and covering heterogeneous domains, MGGCL consistently delivers state-of-the-art performance. Its motif-guided view construction (i) exploits abundant higher-order structures in dense graphs, (ii) uncovers latent micro-clusters in ultra-sparse graphs, and (iii) alleviates popularity bias where star motifs dominate. These results underscore the general efficacy of explicitly modelling higher-order interaction semantics for robust recommendation.

5.5.5 Parameter Sensitivity Analysis

In this section, we investigate the impact of the contrastive learning coefficient λ and the motif ratio θ on MGGCL's performance.

Impact of Contrastive Learning Coefficient λ

Figure 5.3 shows that the contrastive learning coefficient λ has a clear impact on the performance of MGGCL. In general, the performance first improves as λ increases and then decreases when λ becomes large. This trend reflects the role of λ in balancing the supervised recommendation objective and the self-supervised contrastive objective.

When λ is too small, the contrastive learning task contributes only weakly to the overall optimisation process. In this case, the model mainly relies on the recommendation loss and cannot sufficiently exploit the additional structural regularisation provided by motif-guided contrastive learning. As λ increases to an appropriate value, the contrastive objective encourages the model to learn more discriminative and robust user and item representations by preserving useful motif-level semantic relationships. This helps

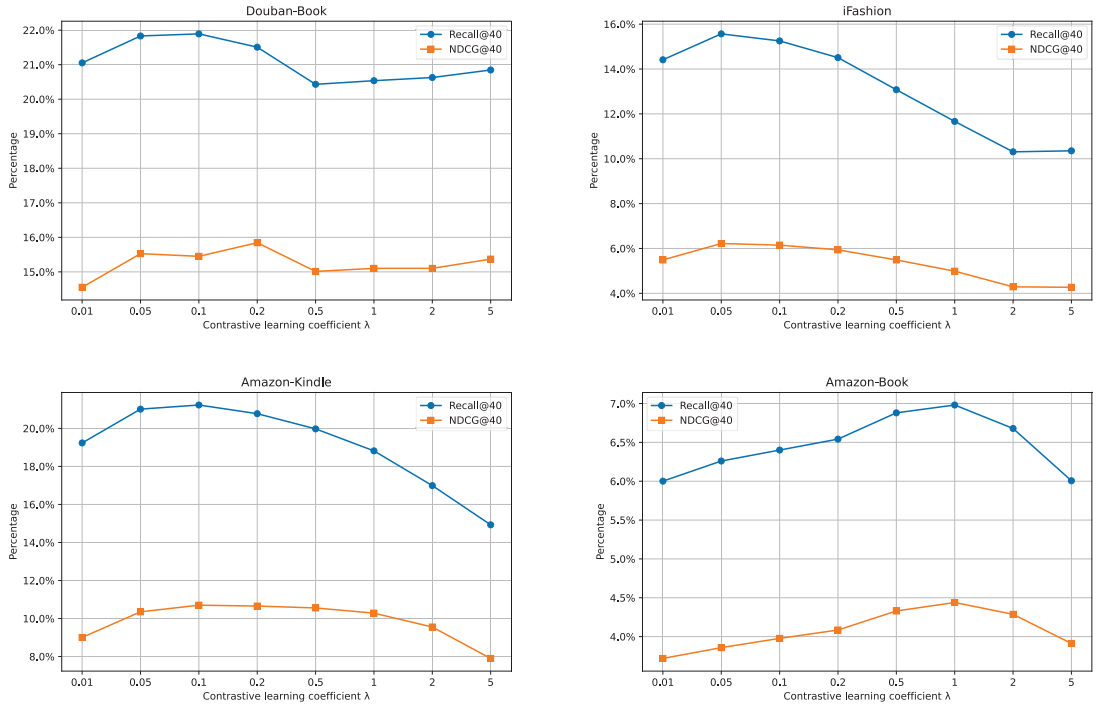


Figure 5.3: Impact of the contrastive learning coefficient λ.

alleviate data sparsity and noise, leading to improved recommendation performance.

However, when λ becomes too large, the contrastive loss may dominate the optimisation process. As a result, the model may over-emphasise representation agreement between contrastive views while paying insufficient attention to the primary recommendation objective. This can weaken the model's ability to capture user preference signals from observed interactions and may introduce over-regularisation. Therefore, the performance declines after the optimal value of λ.

The optimal value of λ differs across datasets. For Douban-Book, iFashion, and Amazon-Kindle, relatively small values of λ achieve the best performance, indicating that a moderate contrastive signal is sufficient to improve representation learning. In contrast, Amazon-Book obtains the best performance when $\lambda = 1$, suggesting that this dataset benefits from a stronger contrastive signal, possibly because the motif-guided contrastive task helps distinguish informative higher-order structures from sparse or noisy interactions.

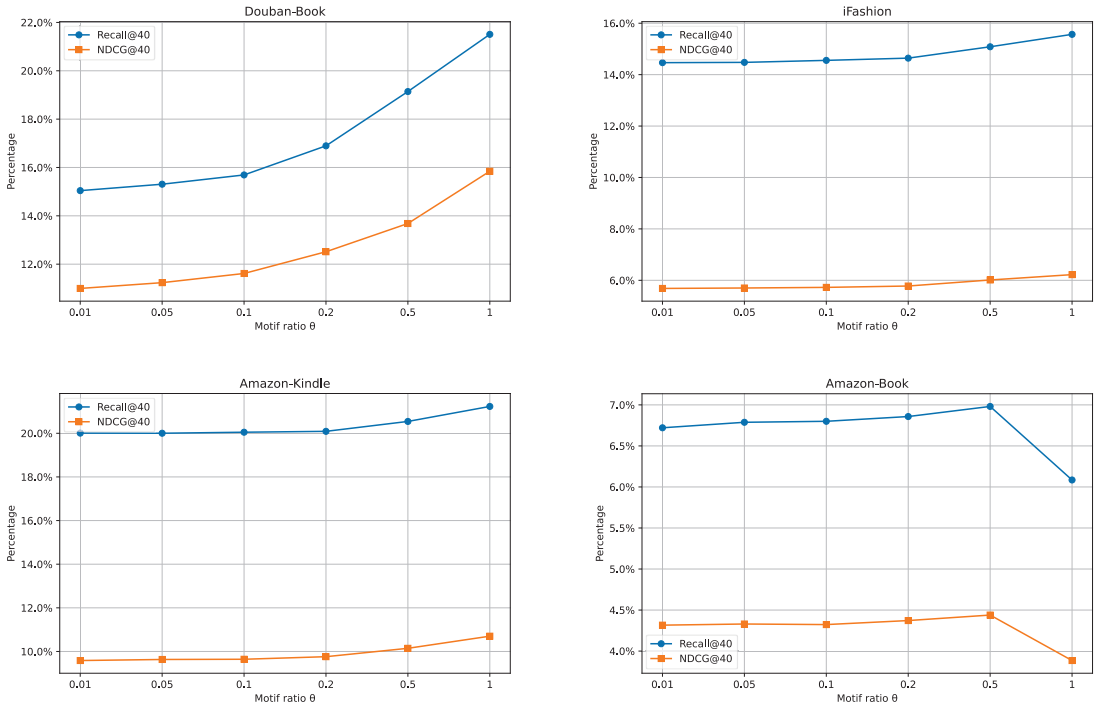


Figure 5.4: Impact of the motif ratio θ.

Impact of Motif Ratio θ

Figure 5.4 illustrates the effect of the motif ratio θ, which controls the relative contribution of the motif adjacency matrices and the original adjacency matrix. The results show that the choice of θ also affects recommendation performance, because it determines how much the model relies on motif-guided structural semantics compared with direct user-item interactions.

When θ is small, the model places greater emphasis on the original adjacency matrix. Although the original graph preserves direct interaction information, it may not sufficiently capture higher-order collaborative patterns encoded by motifs. Therefore, the model may fail to fully exploit the semantic relationships among users and items.

As θ increases, motif-guided information contributes more strongly to representation learning. This allows the model to capture richer structural patterns, such as higher-order co-occurrence and collaborative relationships, which can improve the quality of learned

embeddings. This explains why $\theta = 1$ achieves the best performance on Douban-Book, iFashion, and Amazon-Kindle, where the motif structures appear to provide useful and reliable semantic information.

However, relying entirely on motif-based structures may not always be optimal. For Amazon-Book, the best performance is obtained when $\theta = 0.5$, suggesting that both the original graph and motif-guided graph provide complementary information. In this case, the original adjacency matrix may retain important direct interaction signals that are not fully captured by motifs, while the motif adjacency matrices provide useful higher-order structural semantics. Therefore, a balanced combination of the two sources leads to better performance.

5.6 Conclusion

In this chapter, we introduced a novel motif-guided graph contrastive learning framework to explicitly integrate structural semantics into the construction of contrastive views for graph-based collaborative filtering. Unlike conventional contrastive learning techniques that rely on random perturbations or uniform edge-drop strategies, MGGCL strategically contrasts views dominated by dense rectangle motif against sparse star motifs. This motif-aware approach allows our model to explicitly capture and differentiate meaningful and robust group behaviours from sparse and potentially noisy user-item interactions.

Extensive experiments conducted on four real-world datasets demonstrated that MGGCL consistently achieves superior performance compared to state-of-the-art baseline methods. These results validate our analysis that explicitly modelling motifs helps distinguish critical interaction patterns and significantly improve recommendation quality.

Chapter 6

Motif-Based Heterophily-Aware Graph Contrastive Learning for Recommendation

6.1 Introduction

Graph-based collaborative filtering (CF) has become one of mainstream approaches for recommender systems, thanks to its ability to encode high-order user–item relations with lightweight propagation and to benefit from self-supervised objectives such as graph contrastive learning (GCL). Early graph CF methods augment matrix factorization with neighbourhood aggregation on the user–item bipartite graph (Ying et al., 2018; Fan et al., 2019; H. Wang et al., 2019; X. Wang, He, Wang et al., 2019), while recent models such as LightGCN(X. He et al., 2020) simplify layer-wise propagation to keep only the essential matrix multiplication on graph Laplacian. In parallel, GCL has emerged as a powerful auxiliary signal, contrasting perturbed views of the interaction graph to regularise embeddings and improve generalisation under data sparsity (K. Zhou et al., 2020; Mao et al., 2021; Z. Pan & Chen, 2021; Cheng, Walker, Zhong & Zhou, 2022).

Despite these advancements, persistent challenges remain, including the development of semantically grounded augmentations for bipartite graphs and the mitigation of mixed homophily levels across users and items within graphs. A growing body of work explores random dropout (J. Wu et al., 2021), noise injection (J. Yu et al., 2022), and debiased objectives for CF-specific GCL (Z. Zhou, Xie et al., 2024), yet there is no consensus on how view construction should respect graph structural semantics.

Beyond pairwise edges, higher-order motifs, such as user-anchored triads linking two items through a user, capture recurring relational patterns that maps different behavioural semantics including co-consumption, substitution, or cross-category exploration. Motif-based CF operationalises these patterns to enrich message passing, to reweight edges, or to supervise representation learning, thereby tackling the sparsity issues and improving robustness to noisy edges. Recent motif-based approaches incorporate subgraph counts or motif-induced adjacencies (G. Wang et al., 2023), use hypergraph formulations to fuse multiple relation types(H. Yuan, Yang & Huang, 2022; Y. Sun et al., 2022), or inject motif-aware attention and contrastive objectives. However, these designs rely on fixed thresholds or precomputed labels, and they seldom adapt motif semantics to the mixed-homophily scenarios common in real-world datasets.

A complementary line of research recognises that real-world graphs rarely exhibit uniform homophily (Pei et al., 2019; J. Zhu et al., 2020; Ma, Liu, Shah & Tang, 2023). Users can connect diverse items, and items can attract dissimilar audiences. Standard neighbourhood aggregation can easily blur node distinguishability in heterophilic graphs when dissimilar node features exchange information with each other and become similar. Heterophily-aware graph representation learning therefore rethinks both propagation strategy (Bo et al., 2021; "Heterophily-Aware Personalized PageRank for Node Classification", 2025) so that neighbour information is either preserved or deliberately flipped, depending on context. While these ideas are well-studied in general GNNs, their systematic integration into recommendation remains limited.

In recommender systems, the semantics of a user-anchored triad (i_1, u, i_2) are inherently of different types. The pair of items can be homophilic i.e. category-consistent, heterophilic i.e. cross-category, or mixed. These motif types can thus guide how to aggregate neighbourhood information depending on the homophily level between nodes. They further determine which augmentations are semantically valid for contrastive learning. Dropping edges that encode strong homophily may be harmful, whereas heterophilic edges can be informative to preserve. In essence, motif types are user-centric and data-dependent, so fixed filters cannot capture such locality. This motivates a framework that supervises contrast at the motif level with soft labels estimated online, and constructs views with awareness of different homophily levels between nodes.

We propose Motif-Based Heterophily-Aware Graph Contrastive Learning (MoHeGCL), a structure-aware framework that unifies motif-type supervision and heterophily-aware training. First, we treat each user-anchored triad (i_1, u, i_2) as a contrasting unit. Soft motif labels are computed online via a quantile-scheduled threshold that gradually refines decision boundaries as training stabilises. To improve label quality in early epochs and in sparse scenarios, we use prior features signals from side information for soft motif labelling. Second, we estimate node-wise motif proportions and apply light-weight row reweighting of the adjacency to personalise aggregation strategy for each node, thereby aligning propagation with its local homophily level. Third, our contrastive objective is type-adaptive. We assign motif-specific temperatures and build a type-aware edge-drop view that keeps homophilic edges while selectively preserving heterophilic ones. Paired with a random noise view, this yields coherent, motif-consistent augmentations. The overall training couples the standard BPR recommendation loss with the motif-type aware contrastive loss, maintaining a moderate model complexity.

Our contributions are as follows:

- We treat each user-anchored triad (i_1, u, i_2) as a contrast unit and assign soft motif labels computed online via a quantile-scheduled threshold. To stabilise early training and improve label quality in sparse data, we further integrate prior features signals from side information for soft motif labelling.
- We estimate node-wise motif proportions and apply row reweighting to personalise aggregation strategy for each node. The contrastive objective is type-adaptive via motif-specific temperatures and features a type-aware edge-drop view that keeps homophilic edges while preserving heterophilic ones.
- The overall model complexity remains efficient, making the framework practical at scale. The proposed framework achieves consistent improvements over all baselines on ProgrammableWeb and MovieLens-1M.

The remainder of this chapter is organised as follows. Section 6.3 outlines the model architecture and formalises the problem. Section 6.4 details the MoHeGCL framework which covers motif-aware labels, and heterophily-aware propagation, the motif-type contrastive objective with type-adaptive temperature, and the complexity analysis. Section 6.5 reports experimental settings, baselines, performance comparison, and ablation studies. Finally, Section 6.6 concludes this chapter.

6.2 Related Work

6.2.1 Graph Contrastive Learning for Collaborative Filtering

Graph Contrastive Learning has emerged as a powerful self-supervised learning paradigm for graph-based recommendation. It focuses on learning robust user and item representations by contrasting multiple views of the interaction graph. This approach addresses data sparsity and improves generalisation without relying on explicit supervision. The

foundational work by Wu et al. (J. Wu et al., 2021) introduced SGL, which creates contrastive views through random node and edge dropout. SGL promotes alignment between embeddings derived from different views, which helps mitigate sparsity and enhance robustness. However, its stochastic augmentation strategy can distort meaningful graph structures and introduce semantic inconsistencies and reducing interpretability.

To address these limitations, later models explored simplified or more efficient augmentation strategies. SimGCL (J. Yu et al., 2022) avoids explicit structural perturbations by injecting Gaussian noise directly into node embeddings, so graph connectivity is preserved while maintaining competitive performance. SelfCF (X. Zhou et al., 2023) shifts the focus to augmentation in the output space by applying contrastive learning directly to the final embeddings. By avoiding alteration on the input graph and reliance on negative sampling, this method offers a more lightweight and efficient training process.

Several models aim to refine the contrastive signal or improve its robustness. BDCL (P. Yu et al., 2024) applies directional perturbations and can preserve sparse behavioural data without random corruption. RecDCL (D. Zhang et al., 2024) introduces both batch-wise and feature-wise contrastive objectives to improve embedding orthogonality. DCCR (R. Zhao et al., 2024) uses a self-adaptive noise mechanism and constructs dual collaborative views to extract high-value samples and enhance recommendation diversity. GGDHSCL (X. Liu et al., 2025) introduces a generative diffusion mechanism coupled with hard negative sampling, which strengthens the contrastive signal rather than modifies view structures.

Recent work has also emphasised improved handling of noisy or uninformative views. SymmetricGCL (C. Zhao et al., 2025) introduces a denoising strategy using symmetric contrastive pairs to mitigate degradation from noisy augmentations. WeightedGCL (Z. Chen et al., 2025) enhances SGL with dynamic view-level weighting that allows more informative contrastive views to contribute more significantly to

learning.

To explore more stable and informative view construction, NLGCL (J. Xu et al., 2025) generates contrastive views from naturally existing neighbour layers instead of random perturbations. This reduces semantic drift while maintaining graph topology. DimCL (C. Zhang et al., 2025) augments representations in a dimension-aware fashion to encourage diverse embedding components for contrastive learning.

To support privacy and personalisation, Personalised Federated Contrastive Learning (S. Wang et al., 2025) adapts GCL to the federated setting by aggregating user-device interactions without centralising user data. This design introduces personalisation and scalability but requires careful calibration to maintain embedding consistency across clients.

These works discussed above illustrate a trajectory from random augmentations toward more principled and robust contrastive learning techniques. However, many still operate with implicit structure and offer limited topological semantics, motivating structurally grounded alternatives.

Recognising the limitations of uniform random perturbations, several researchers have explored structured or semantically informed augmentations. SG-CLR (Y. Zhang et al., 2024) enhances the semantic integrity of contrastive views by guiding edge additions based on homophily. HGCL (J. Xue et al., 2025) takes a hierarchical perspective by clustering items into multiple resolution layers to preserve coarse-to-fine structural semantics. NCL (Z. Lin et al., 2022) advances the idea of local structure by treating topological neighbours as natural positives rather than sampling corrupted ones. This strategy captures contextual consistency and reduces semantic drift during augmentation.

JCG (Dai et al., 2024) combines structural and semantic contrastive objectives to address data imbalance and reduce the influence of high-degree nodes. This improves generalisation across heterogeneous graph distributions. MD-GCCF (X. Li et al.,

2024) advances multi-view learning by integrating deep collaborative signals and designing both local and global contrastive objectives. By mitigating over-smoothing and reducing redundancy, this architecture yields more informative representations for recommendation.

Several models introduce learnable or deterministic transformation schemes. CL-KDM (Ni et al., 2024) bypasses traditional dropout strategies entirely by maximising statistical dependence between views, which preserves global latent geometries without edge removal. LMACL (X. Liu et al., 2024) combines GCN and GAT to learn augmentation strategies to generate multi-hop and neighbour-aware contrastive views.

These methods present promising steps toward augmentations that encode high-quality semantics or structure. However, many rely on heuristics or predefined rules and rarely integrate clear higher-order graph patterns such as motifs into the view generation process.

6.2.2 Motif-Based Graph Neural Networks for Recommender Systems and Contrastive Learning

Recent studies have explored motif-based enhancements in recommender systems. Motifs-Res (Y. Sun et al., 2022) formulates motif co-occurrence as hyperedges and applies hypergraph convolution with contrastive loss to exploit local motif semantics. MotifSR (Cui et al., 2021) incorporates motif-enhanced temporal subgraphs to better capture session-level user behaviours in sequential recommendation. These models demonstrate the growing interest in exploiting motif semantics to strengthen signal quality and model generalisation in graph-based recommendation.

Complementing this, motif-based contrastive learning has been studied in other domains. Motif-driven contrastive learning of graph representations (S. Zhang et al., 2024) introduces motif-induced views to enforce global and local consistency in general

graph representation learning, where motif co-occurrence defines semantically consistent neighbourhoods. MotifCC (X. Wu et al., 2024) proposes a framework that builds subgraphs from triangle motifs and applies contrastive learning between lower-order and higher-order views to jointly preserve structural uniqueness and encourage clustering separability. MHGCL (C. Li et al., 2025) jointly leverages motif structures and node homogeneity to refine the selection of contrastive pairs. Therefore, it suppresses noise and improves representation discrimination in general graph tasks. Although effective, these works do not target recommender systems and typically operate in node classification or community detection settings.

6.2.3 Heterophily-Aware Graph Neural Networks for Recommender Systems

Most GNNs are builted under the assumption of homophily (X. Zheng et al., 2022), where similar nodes are more likely to connect to each other while dissimilar nodes are less likely to be adjacent. In contrast, graph heterophily is when dissimilar nodes are more likely to be connected. When working on some real world datasets that exhibit heterophily rather than homophily, many of these models face the challenge of performance degradation. To address this limitation, a series of research efforts have been made on specifically designed GNNs for heterophilic graphs.

Fabbri et al. (Fabbri et al., 2022) introduce an agent–recommender co-evolution simulator that varies group size, homophily level, and link-recommendation strategies to study long-term exposure dynamics. The study shows that when a minority is homophilic it gains disproportionate visibility, whereas a heterophilic minority becomes underexposed. Futhermore, link recommenders strengthen individual-level exposure inequality independent of group homophily. This evidence motivates heterophily-aware modeling in recommendation, where algorithms explicitly account for how mixing

patterns interact with learning and feedback loops.

Peng et al. (S. Peng et al., 2022) propose a spectral reweighting framework that emphasizes informative graph-frequency components for collaborative filtering. By preserving high-frequency signals that typically encode cross-community or dissimilar-neighbour information, the method mitigates over-smoothing and retains predictive cues characteristic of heterophilic neighbourhood. Frequency-selective propagation operates as a practical surrogate for heterophily-awareness when dissimilar relations carry complementary information.

Jiang et al. (W. Jiang et al., 2024) introduce SHaRe, a data-centric social recommendation framework that diagnoses preference-aware homophily and rewires the social graph by adding highly aligned ties while pruning heterophilic ones. To address representation drift during rewiring, SHaRe integrates a contrastive calibration objective. As a result, it improves accuracy across homophily ratios and enhances a range of graph-based recommenders. This line of work share a similar strategy of heterophily-awareness that models measure local alignment and then modify structure and training signals accordingly.

In (Gholinejad & Chehreghani, 2024), the authors present HetroFair, a fairness-oriented GNN recommender that combines fairness-aware attention with heterophily feature weighting to assign distinct aggregation weights under heterophilous conditions. Experiments across multiple datasets indicate simultaneous gains in item-side fairness and user-side accuracy, which shows the abilities of retaining informative dissimilar neighbours and mitigating popularity-driven exposure disparities.

Despite these advances, existing methods still exhibit research gaps. Some rewire graphs toward latent homophily (W. Jiang et al., 2024) while others rely on frequency selection that implicitly preserves heterophilous cues (S. Peng et al., 2022). In addition, some methods couple propagation with fairness controls (Gholinejad & Chehreghani, 2024). These methods offer limited mechanisms to explicitly supervise and exploit

heterophily at fine-grained structural units. In particular, existing social-exposure analyses (Fabbri et al., 2022) highlight the importance of these effects but do not provide a motif-level training signal that distinguishes homophilic, mixed, and heterophilic interactions during contrastive learning and propagation. To address this gap, we propose MoHeGCL to treat user-anchored triads as contrast units with soft motif-type labels and employ a motif-conditioned training strategy to leverage heterophilic structure for recommendation.

6.3 Model Overview and Problem Setup

We study top-K recommendation from implicit feedback under mixed homophily and heterophily signals. Let U and I denote the sets of users and items, with $|U| = n_u$, $|I| = n_i$. The observed interactions are $R \subseteq U \times I$, where an edge $(u, i) \in R$ indicates user u interacted with item i. We build a bipartite graph $G = (V, E)$ with $V = U \cup I$ and $E = \{(u, i) \mid (u, i) \in R\}$. The goal is to learn embeddings $\mathbf{p}_u, \mathbf{q}i \in \mathbb{R}^d$ that rank items by $\hat{y}_{ui} = \mathbf{p}_u^\top \mathbf{q}_i$ and generate top-K recommendations.

Each item i is associated with one or more labels $c(i) \in \mathcal{C}$ derived from side information including categories and tags. For a user u and two items i_1, i_2 with $(u, i_1), (u, i_2) \in R$, the length-2 bipartite walk (i_1, u, i_2) forms an item–user–item triad. We partition triads into three semantic channels using the relationship between $c(i_1)$ and $c(i_2)$:

- Homophilic triad: $c(i_1)$ and $c(i_2)$ are the same or strongly overlapping.
- Heterophilic triad: $c(i_1)$ and $c(i_2)$ are disjoint.
- Mixed triad: $c(i_1)$ and $c(i_2)$ share partial overlap.

Figure 6.1 illustrates an example of the three types of triads. User u_1 is connected with item i_1 and i_2, which both have labels of class 1 and 2, so triad (i_1, u_1, i_2) is a

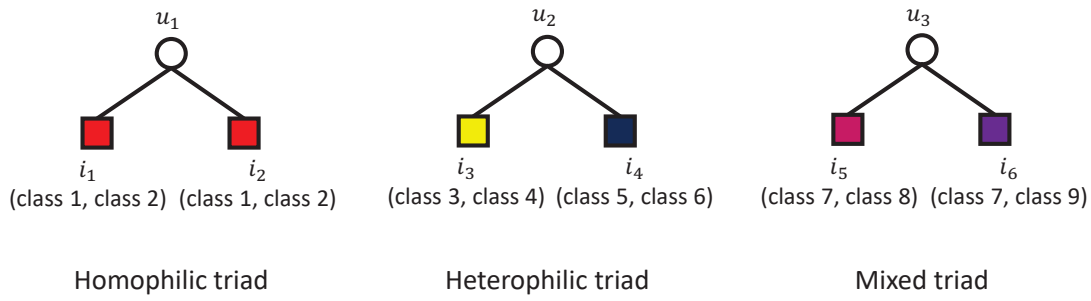


Figure 6.1: Illustration of an example of homophilic, heterophilic, and mixed triads.

homophilic triad. User u_2 is connected with item i_3 and i_4, which have distinctive labels, so triad (i_3, u_2, i_4) is a heterophilic triad. User u_3 is connected with item i_5 and i_6, which share one common label of class 7 but have other different labels, so triad (i_5, u_3, i_6) is a mixed triad.

This partition yields three item–item motif graphs $M^{\text{homo}}, M^{\text{mixed}}, M^{\text{hetero}}$. From these we induce three normalized propagation operators on the bipartite graph, denoted $\tilde{\mathbf{A}}^{\text{homo}}, \tilde{\mathbf{A}}^{\text{mixed}}, \tilde{\mathbf{A}}^{\text{hetero}}$. Intuitively, homophilic triad emphasizes within-type coherence, heterophilic triad emphasises cross-type discovery, and mixed triad captures soft semantic proximity.

We use a standard LightGCN encoder on the user–item bipartite graph. Let $\tilde{\mathbf{A}}$ be the single, normalised adjacency matrix. Starting from initial embeddings $\mathbf{E}^{(0)} = [\mathbf{P}^{(0)}; \mathbf{Q}^{(0)}]$, we propagate for L layers without MLPs or feature transformations:

$$\mathbf{E}^{(l+1)} = \tilde{\mathbf{A}}, \mathbf{E}^{(l)}, \qquad l = 0, \ldots, L-1. \tag{6.1}$$

The final representation is the layer-wise mean

$$\mathbf{E} = \frac{1}{L+1} \sum_{l=0}^{L} \mathbf{E}^{(l)}, \tag{6.2}$$

which we split into user and item blocks $\mathbf{P}, \mathbf{Q}$. All motif awareness is handled outside the backbone via the view construction and the contrastive supervision.

MoHeGCL employs two complementary views of the same graph signal:

- View-α: A standard LightGCN forward pass with uniform random perturbations on node embeddings.
- View-β: For each user, we reweight the adjacency matrix according to the fractions of homophilic, heterophilic, and mixed triads for each user.

View construction details are specified in Section 6.4.3.

To supervise contrastive pairs with motif type aware augmentations, MoHeGCL derives soft motif labels for item–item pairs and schedules a quantile threshold over training to gradually harden positives while mitigating early noise. It also employs type-adaptive temperatures $\theta_{\text{homo}}, \theta_{\text{mixed}}, \theta_{\text{hetero}}$ to balance gradients across channels with different density and difficulty.

6.4 Method

In this section, we present the architectural details of MoHeGCL. Algorithm 4 summarises the overall training flow. Implementation details for the features as prior, quantile schedule and motif labelling are provided in Section 6.4.1, and the layer propagation and contrastive objective are described in Section 6.4.2 and 6.4.3 respectively.

6.4.1 Motif-Aware Labelling

The contrastive learning is supervised by assigning soft motif-aware labels to item–item pairs. To generate these labels, we combine prior features from side information and learned node embeddings, which are then selected by a quantile scheduled threshold.

Algorithm 4: MoHeGCL

Input: Interaction graph G, epochs T, batch size B, LightGCN layers L, random noise ϵ, contrastive learning weight λ_{cl}, quantile schedule $q(t)$, type temperatures $\theta_{\text{homo}}, \theta_{\text{mix}}, \theta_{\text{hetero}}$, keep probabilities $(p_{\text{homo}}, p_{\text{mix}}, p_{\text{hetero}})$, prior features $\hat{e}_i$ (items) and $\hat{e}_u$ (users).

Output: Final node embeddings for inference

Initialize trainable user/item embeddings and load feature priors $\hat{e}_u, \hat{e}_i$

for $t \leftarrow 1$ **to** T **do**

- compute no-grad embeddings $(E_u^{\text{snap}}, E_i^{\text{snap}}) \leftarrow \text{LightGCN}(G)$
- sample a subset of item pairs and compute similarities $s(i, i') = \cos(E_i^{\text{snap}}, E_{i'}^{\text{snap}})$
- estimate quantile-based thresholds $(\tau_{\text{lo}}, \tau_{\text{hi}}) \leftarrow Q_{q(t)}(\{s(i, i')\})$
- estimate per-user motif fractions $\hat{h}_u = (\hat{h}^{(\text{homo})}, \hat{h}^{(\text{mix})}, \hat{h}^{(\text{hetero})})$
- set keep probability $p_{\text{keep}}(u) \leftarrow \sum_k \hat{h}_u^{(k)} p_k$
- construct row-scaled adjacency A^β using $p_{\text{keep}}(u)$
- compute cached View-β embeddings $(E_u^\beta, E_i^\beta) \leftarrow \text{LightGCN}(A^\beta)$ (stop-grad)
- **foreach** *mini-batch of interactions* **do**
  - compute View-α embeddings (LightGCN forward + random noise ϵ)
  - compute BPR loss and L_2 regularisation on batch
  - sample triads (i_1, u, i_2) from user histories
  - compute item–item similarity $s(i_1, i_2) = \lambda \cos(\hat{e}_{i_1}, \hat{e}_{i_2}) + (1 - \lambda) \cos(e_{i_1}, e_{i_2})$
  - compute soft motif weights $w = (w_{\text{homo}}, w_{\text{mix}}, w_{\text{hetero}})$ from $s(i_1, i_2)$ and $(\tau_{\text{lo}}, \tau_{\text{hi}})$ using sigmoid functions; normalize to $\tilde{w}$
  - set per-sample temperature $\theta(m) \leftarrow \tilde{w}_{\text{homo}}\theta_{\text{homo}} + \tilde{w}_{\text{mix}}\theta_{\text{mix}} + \tilde{w}_{\text{hetero}}\theta_{\text{hetero}}$
  - compute two-view supervised contrastive loss L_{cl} between View-α and View-β using $\theta(m)$
  - total loss $L \leftarrow L_{\text{rec}} + \lambda_{\text{reg}} \|\Theta\|_2^2 + \lambda_{\text{cl}} L_{\text{cl}}$
  - backpropagate and update parameters (Adam)
- **end**

end

return final LightGCN embeddings (E_u, E_i)

Features as Prior

We incorporate observed item categories as prior features that initialises representation learning and stabilises early optimisation. Let x_i denote the raw feature vector of item i. We first generate a multi-hot embedding from the item's categories, then project this vector to a low-dimensional space.

We construct priors from side information and explicitly stop gradients so they never receive supervision signals. A small projector $W \in \mathbb{R}^{d\times \dim(x)}$ maps features into the model space, and we freeze a copy for labelling:

$$\tilde{\mathbf{e}}_i = \mathrm{stopgrad}(Wx_i), \qquad \tilde{\mathbf{e}}_u = \mathrm{Mean}\left(\{\tilde{\mathbf{e}}_i, i \in \mathcal{N}(u)\}\right). \tag{6.3}$$

These priors provide a stable and low-variance signal to guide early label construction. They are not used in forward prediction and are never updated by backpropagation.

In contrast to the frozen feature priors, the trainable pathway learns embeddings purely from the graph structure and propagates them with a LightGCN backbone. We use Xavier initialisation for user embeddings $P \in \mathbb{R}^{|\mathcal{U}|\times d}$ and item embeddings $Q \in \mathbb{R}^{|\mathcal{I}|\times d}$. Let $\hat{A}$ denote the precomputed row-normalised user–item adjacency matrix. At each forward pass we concatenate $E^{(0)} = [P; Q]$ and apply L layers of linear propagation,

$$E^{(\ell)} = \hat{A}E^{(\ell-1)} \quad (\ell = 1, \ldots, L), \tag{6.4}$$

We then average the propagated layers only, i.e., $\frac{1}{L}\sum_{\ell=1}^{L} E^{(\ell)}$, to obtain the final representations $\bar{E} = \left[\mathbf{E}_u, \mathbf{E}_i\right]$, which are split back into user and item embeddings. The encoder exposes: (i) a base view for standard scoring; (ii) an efficient noisy view produced from cached base embeddings by adding sign-aligned, ℓ_2-normalised noise without re-running sparse propagation; and (iii) a row-scaled view in which user rows of $\hat{A}$ are re-weighted according to motif labels before propagation. Predictions use the

dot product $\langle \mathbf{E}_u, \mathbf{E}_i \rangle$, and the embeddings are optimised end-to-end with BPR and ℓ_2 regularisation. This design use prior features to inform label construction only, while trainable embeddings are the sole parameters updated by the training loss and used for recommendation.

Soft Motif Labelling and Quantile-Scheduled Threshold

We score each triad $m = (i_1, u, i_2)$ by the semantic similarity between its two items. To stabilise early training while allowing the learnable embeddings to take over later, we combine a prior item similarity with a representation item similarity:

$$s(i_1, i_2) = \lambda \cos\big(\tilde{\mathbf{e}}_{i_1}, \tilde{\mathbf{e}}_{i_2}\big) + (1 - \lambda) \cos\big(\mathbf{e}_{i_1}, \mathbf{e}_{i_2}\big), \tag{6.5}$$

where $\lambda : 1 \to 0$ gradually decreases over $T \in [5, 10]$ epochs. Early epochs rely on side-information priors $\tilde{\mathbf{e}}_i$, while later epochs progressively trust the learned item embeddings $\mathbf{e}_i$.

Given a batch of triads, we form a sampled distribution of item–item similarities $\{s(i, i')\}$ without calculating for all $|I|^2$ pairs. For each user u we subsample up to K interacted items and compute all within-user pairs, generating $O(BK^2)$ pairs per batch. From this set we draw quantile-scheduled thresholds

$$\tau_{\text{low}} = Q_{q_t/2}, \qquad \tau_{\text{high}} = Q_{1-q_t/2}, \tag{6.6}$$

where q_t increases over training, sharpening the decision boundary. Intuitively, early q_t is conservative so many pairs count as roughly homophilic, while later q_t raises τ_{high} and lowers τ_{low} to separate classes more clearly.

For a triad $m = (i_1, u, i_2)$ with similarity $s = s(i_1, i_2)$, we assign soft motif type via

sigmoids:

$$\begin{aligned} w_{\text{homo}} &\propto \sigma\Big(\tfrac{s-\tau_{\text{high}}}{\gamma}\Big), \gamma > 0 \\ w_{\text{hetero}} &\propto \sigma\Big(\tfrac{\tau_{\text{low}}-s}{\gamma}\Big), \\ w_{\text{mixed}} &= 1 - w_{\text{homo}} - w_{\text{hetero}}. \end{aligned} \tag{6.7}$$

These scores are differentiable in s and normalised to probabilities $\tilde{w}_{\text{homo}}, \tilde{w}_{\text{mixed}}, \tilde{w}_{\text{hetero}}$. In practice, s is computed by broadcasting item embeddings within each sampled triad. No user–item similarity is used for labelling.

The soft labels serve two roles. Firstly, we estimate each user's motif fractions, i.e. homophilic, heterophilic, or mixed, from a small sample of their triads and normalise them for each user. These fractions modulate the each user's keep probability when constructing the type-aware View-β, which encourages feature alignment for homophily-dominant users and feature separation for heterophily-dominant users. Secondly, the soft motif labels are used to calculate a triad specific temperature for the contrastive loss.

This combination provides stable heterophily-aware supervision with dynamically estimated thresholds, while remaining meaningful to motif semantics rather than user–item preference strength. The quantile-scheduled $(\tau_{\text{low}}, \tau_{\text{high}})$ is estimated efficiently by random sampling. The sigmoid-based soft motif labels are generated from item–item semantics.

6.4.2 Heterophily-Aware Motif Propagation

We adapt LightGCN propagation so that the strength of message passing is user-wise gated by the mix of motif types inferred dynamically. Let $\tilde{\mathbf{A}}$ be the row-normalized bipartite adjacency used by LightGCN, and $\mathbf{E}^{(0)} = [\mathbf{E}_u^{(0)}; \mathbf{E}_i^{(0)}]$ be the trainable user/item embeddings. The base LightGCN backbone performs L layers of linear propagation with layer-wise averaging:

$$\mathbf{E}^{(\ell+1)} = \tilde{\mathbf{A}}\mathbf{E}^{(\ell)}, \qquad \overline{\mathbf{E}} = \frac{1}{L}\sum_{\ell=1}^{L}\mathbf{E}^{(\ell)}, \quad (\mathbf{E}_u, \mathbf{E}_i) = \mathrm{split}(\overline{\mathbf{E}}). \tag{6.8}$$

This base view is implemented as a standard matrix multiplication stack, with neither feature transformation or non-linearity.

To decide whether performing feature alignment or separation, we construct a row-scaled adjacency matrix according to user's prevailing motif type,

$$\tilde{\mathbf{A}}^{\beta} = \mathrm{diag}(\mathbf{p})\tilde{\mathbf{A}}, \tag{6.9}$$

and run the same LightGCN propagation on $\tilde{\mathbf{A}}^{\beta}$ to obtain a second view View-β, i.e. $(\mathbf{E}_u^{\beta}, \mathbf{E}_i^{\beta})$. Here $\mathbf{p} \in [0,1]^U$ is a user specific keep probability that approximates type-aware edge drop by uniformly scaling all outgoing messages of that user while the item rows are left unscaled. This operation is implemented efficiently by multiplying the sparse adjacency matrix with row-specific factors.

At the start of each epoch, we first take a no-grad snapshot of the current base embeddings $\mathbf{E}_u^{\text{snap}}, \mathbf{E}_i^{\text{snap}}$. Then we estimate a global similarity quantile threshold τ from a large random sample of user–item pairs which serves as the moving threshold for label softening. Lastly, we compute user specific motif fractions $\hat{h}_{\text{homo}}(u), \hat{h}_{\text{mix}}(u), \hat{h}_{\text{hetero}}(u)$ by sampling small item pairs from each user's history and applying the soft motif labelling mechanism from Section 6.4.1. These fractions are then linearly combined with three keep probabilities $p_{\text{homo}}, p_{\text{mix}}, p_{\text{hetero}} \in [0,1]$ for homophilic, mixed, and heterophilic triads respectively. We set them static to $p_{\text{homo}}, p_{\text{mix}}, p_{\text{hetero}} = (0.9, 0.7, 0.5)$, which encodes a empirical prior of homophilic > mixed > heterophilic, preserving dense connections for propagation, moderately weakening potentially noisy edges, and applying stronger feature separation to the distinctive neighbours. It maintains connectivity and stable training while increasing view diversity where it matters.

$$p_u = p_{\text{homo}}\hat{h}_{\text{homo}}(u) + p_{\text{mix}}\hat{h}_{\text{mix}}(u) + p_{\text{hetero}}\hat{h}_{\text{hetero}}(u), \tag{6.10}$$

which forms the user-row scaler $\mathbf{p} = (p_u)_{u=1}^{U}$ for View-β in that epoch. Intuitively, if a user's interaction triads are more heterophilic, a larger p_{hetero} will highlight feature separation, while homophily-dominant users can get more feature alignment via a larger p_{homo}.

6.4.3 Contrastive Objective

For each sampled triad $m = (u, i_1, i_2)$, we build a composite representation by pooling the user and two item embeddings from a given view. In our implementation the pooling is a simple concatenation

$$\phi(\mathbf{e}_u, \mathbf{e}_{i_1}, \mathbf{e}_{i_2}) = [\mathbf{e}_u | \mathbf{e}_{i_1} | \mathbf{e}_{i_2}], \tag{6.11}$$

which preserves neighbour roles without extra parameters.

We contrast triad embeddings from two synchronised views per mini-batch:

- View-α: Starting from the current base LightGCN embeddings, we apply random sign-guided L_2-normalised noise to users and items, then pool triads: $\mathbf{z}_m^{\alpha} = \phi(\tilde{\mathbf{e}}_u, \tilde{\mathbf{e}}_{i_1}, \tilde{\mathbf{e}}_{i_2})$. This view backpropagates gradients.

- View-β: Once per epoch we take a no-grad snapshot of embeddings produced by propagating on a row-scaled normalised adjacency matrix $\tilde{\mathbf{A}}^{\beta} = \text{diag}(\mathbf{p})\tilde{\mathbf{A}}$. The diagonal matrix $\text{diag}(\mathbf{p})$ scales user rows by per-user keep probabilities p_u. This approximates type-aware edge-drop while keeping propagation efficient. The resulting snapshot embeddings $(\mathbf{e}_u^{\beta}, \mathbf{e}_i^{\beta})$ are frozen within the epoch and pooled into $\mathbf{z}_m^{\beta} = \phi(\mathbf{e}_u^{\beta}, \mathbf{e}_{i_1}^{\beta}, \mathbf{e}_{i_2}^{\beta})$.

Let m be a triad in the batch $\mathcal{B}$. We use a minimal supervised-contrastive objective with in-batch negatives and the diagonal pair $(\mathbf{z}_m^{\alpha}, \mathbf{z}_m^{\beta})$ as the sole positive of anchor m:

$$\mathcal{L}_{\text{cl}} = \sum_{m \in \mathcal{B}} -\log \frac{\exp\left(\langle \hat{\mathbf{z}}_m^{\alpha}, \hat{\mathbf{z}}_m^{\beta} \rangle / \theta(m)\right)}{\sum_{m' \in \mathcal{B}} \exp\left(\langle \hat{\mathbf{z}}_m^{\alpha}, \hat{\mathbf{z}}_{m'}^{\beta} \rangle / \theta(m)\right)}, \tag{6.12}$$

where $\hat{\mathbf{z}} = \mathbf{z}/||\mathbf{z}||_2$. The triad specific temperature $\theta(m) = \tilde{w}_{\text{homo}}\theta_{\text{homo}} + \tilde{w}_{\text{mixed}}\theta_{\text{mix}} + \tilde{w}_{\text{hetero}}\theta_{\text{hetero}}$, which scales the contrastive logits for the paired-view SupCon loss with diagonal positives. For numerical stability we subtract the per-row maximum from logits before exponentiation.

Alongside $\mathcal{L}_{\text{cl}}$ we train with standard pairwise BPR on the current base embeddings:

$$\mathcal{L}_{\text{rec}} = -\sum_{u,i,j \in \mathcal{B}} \log(\sigma(\hat{y}_{ui} - \hat{y}_{uj})), \tag{6.13}$$

where $\mathcal{B}$ is the sampled batch, σ is the sigmoid function, $\hat{y}_{ui}$ is the inner product of user and item embeddings, i and j indicate positive and negative items respectively.

The total loss per batch is

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \lambda_{\text{reg}}||\Theta||_2^2 + \lambda_{\text{cl}}\mathcal{L}_{\text{cl}}. \tag{6.14}$$

where Θ is the set of all model parameters, hyperparameters λ_{reg} and λ_{cl} controls the magnitudes of L_2 regularisation and contrastive loss respectively.

6.4.4 Complexity Analysis

In this subsection, we analyse the complexity of MoHeGCL in comparison with SimGCL, which is a strong and light weight graph contrastive learning baseline. Since MoHeGCL only introduces a small projection matrix for feature priors apart from the standard user and item embeddings, the space complexity is of the same order as SimGCL. The inference time complexity is also the same because test propagation is

identical.

Let $|V|$ and $|E|$ be the numbers of nodes and edges in the user–item graph, s be the number of epochs, B be the batch size, d be the embedding size, L be the number of layers, and S be the number of sampled user–item pairs per epoch used by MoHeGCL's quantile estimator. In our experiment settings, $d = 64, L = 2, B = 2048, S = 2 \times 10^6$. Sparse propagation costs $O(L|E|d)$. The training cost is dominated by the following components:

- Backbone propagation: For both MoHeGCL and SimGCL, each epoch performs $O(L|E|d)$ from sparse matrix multiplication in the batch loop.
- Within-batch contrastive head: SimGCL computes pairwise logits over a $B \times B$ similarity matrix between two views. MoHeGCL does likewise over triad-pooled features which has a constant 3 times of width. Per batch the complexity is $O(B^2 d)$, and for $|E|/B$ batches per epoch it aggregates to $O(|E|Bd)$.
- Additionally, MoHeGCL performs once per epoch: 1. Quantile-scheduled thresholding on S sampled similarities: $O(Sd)$. 2. Lightweight row reweighting of the adjacency during construction of View-β graph: $O(|E|)$. 3. One extra propagation on the reweighted adjacency matrix to cache the type-aware embeddings for the entire epoch: $O(L|E|d)$.

Aggregating all the components, SimGCL has a total per-epoch complexity of:

$$O\big(L|E|d\big) + O\big(|E|Bd\big), \tag{6.15}$$

For MoHeGCL, since $O(|E|) \subseteq O(L|E|d)$, we have omitted the $O(|E|)$ and the second $O(L|E|d)$ and obtain the final complexity as:

$$O\big(L|E|d\big) + O\big(|E|Bd\big) + O\big(Sd\big). \tag{6.16}$$

Under the standard recommender-system settings used in our experiments, MoHeGCL retains SimGCL's per-epoch scaling $O\big(L|E|d\big) + O\big(|E|Bd\big)$. The additional $O(Sd)$ for quantile-scheduled thresholding and an extra layer propagation per epoch introduce only a small constant-factor overhead, while enabling MoHeGCL's motif-aware supervision.

6.5 Experiments

6.5.1 Experimental Settings

We conduct experiments on two representative datasets: ProgrammableWeb and MovieLens-1M. Both datasets provide side information that we leverage to generate user and item embeddings used as feature priors. For ProgrammableWeb, each mashup (user) and API (item) is annotated with multiple categorical tags. We encode these tags using a shared label-embedding and pooling method. Specifically, each unique label is represented by a learnable 64-dimensional vector, and each node's feature vector is the mean of its label embeddings, forming a bag-of-labels representation. The resulting embeddings serve as feature priors for both users and items.

For MovieLens-1M, only movies have genres as categorical metadata. We first map each movie's genres to a multi-hot vector and embed them with the same label-embedding and pooling mechanism as above, obtaining 64-dimensional genre embeddings. User embeddings are then computed as the average of the genre embeddings of the movies they have rated. These derived features act as priors for user representations. Both datasets are randomly divided into training and test sets in a 4:1 ratio. For evaluation, we employ two widely used top-k ranking metrics: Recall@k and NDCG@k, where $k \in \{5, 10, 20, 40\}$.

6.5.2 Baselines and Hyperparameter Settings

We compare our method against four strong baselines as follows.

- LightGCN(X. He et al., 2020): Standard message passing on the user–item bipartite graph without nonlinearities or feature transformations. Final representations are the layer-wise mean.
- SelfCF(X. Zhou et al., 2023): Self-supervised CF that contrasts representations without explicit graph augmentations. We follow the original configuration of two stochastic views from representation perturbations on the same LightGCN encoder.
- SGL(J. Wu et al., 2021): Self-supervised Graph Learning with three choice of graph augmentations, including edge-dropout, node-dropout, and random-walk. We use edge-dropout for the experiments due to its better performance.
- SimGCL(J. Yu et al., 2022): Simple Graph Contrastive Learning with random noise augmentation on view embeddings.

All baselines share the same optimiser, negative sampling, and evaluation stack as our model. The hyperparameters of all baseline methods are tuned according to their original papers, except that all models have the following same settings. Embeddings size is 64, the batch size is 2048, and the number of MLPs or graph convolution layers is two.

For MoHeGCL, we tune the following hyperparameters as follows. Contrastive learning weight λ_{cl} is searched from $\{0.01, 0.1, 0.5, 1\}$; random noise scale is searched from $\{0.01, 0.1, 0.5\}$; learning rate is searched from $\{0.001, 0.01, 0.1\}$; L_2 regularisation weight λ_{reg} is searched from $\{0.00001, 0.0001, 0.001\}$; the initialisations of type-adaptive temperatures θ_{homo}, θ_{mix}, θ_{hetero} are grid-searched from $\{0.1, 0.2, 0.5\}$.

Table 6.1: Performance comparison of baseline models and MoHeGCL on the ProgrammableWeb and MovieLens-1M datasets.

Dataset	Model	Recall@5	NDCG@5	Recall@10	NDCG@10	Recall@20	NDCG@20	Recall@40	NDCG@40
ProgrammableWeb	LightGCN	0.38223	0.31647	0.47301	0.34774	0.58477	0.37921	0.68068	0.40273
	SGL	0.37645	0.31821	0.46878	0.35045	0.56474	0.37844	0.63102	0.39509
	SelfCF	0.34539	0.30318	0.45059	0.33904	0.52753	0.36161	0.62212	0.38433
	SimGCL	0.40219	0.33334	0.48266	0.36042	0.5755	0.38792	0.67618	0.41191
	MoHeGCL	**0.41085**	**0.34973**	**0.51105**	**0.38336**	**0.60382**	**0.41054**	**0.7054**	**0.43544**
MovieLens-1M	LightGCN	0.11041	0.32503	0.17757	0.30554	0.27065	0.30697	0.39118	0.33517
	SGL	0.11253	0.33566	0.17795	0.31265	0.26606	0.31003	0.37941	0.33403
	SelfCF	0.08013	0.25305	0.12625	0.23305	0.19554	0.23146	0.29468	0.25417
	SimGCL	0.11678	0.33143	0.18814	0.31473	0.28281	0.3166	0.40444	0.34546
	MoHeGCL	**0.12093**	**0.34254**	**0.19204**	**0.32289**	**0.28669**	**0.32394**	**0.40929**	**0.35288**

6.5.3 Performance Comparison

Table 6.1 shows the experiment results. Overall, MoHeGCL outperform all baseline methods across the two datasets.

First, across both datasets and all cut-offs, MoHeGCL attains the best Recall@K and NDCG@K, indicating consistent gains in both recommendation coverage and ranking quality. For ProgrammableWeb dataset, MoHeGCL surpasses the strongest baseline SimGCL by 2.15% in Recall@5 and 4.92% in NDCG@5. The relative improvements grow at moderate cut-offs, reaching 5.88% in Recall@10 and 6.36% in NDCG@10, and remain robust at larger K, e.g., 4.32% in Recall@40 and 5.71% in NDCG@40. Compared with the non-contrastive LightGCN, MoHeGCL yields larger NDCG gains, including 10.51% in NDCG@5 and 8.26% in NDCG@20, suggesting that motif-type supervision particularly enhances the ordering of relevant items near the top of the list.

For the MovieLens-1M dataset, improvements are smaller but still uniform. Relative to SimGCL, MoHeGCL improves by 3.55% in Recall@5 and 3.35% in NDCG@5, with positive margins maintained at all K, e.g., 2.59% in NDCG@10, 2.32% in NDCG@20, 2.15% in NDCG@40. Against LightGCN, MoHeGCL achieves 9.53% in Recall@5 and 5.39% in NDCG@5. The more modest improvements on MovieLens-1M align with its stronger homophily. When user–item relations are more homophilic, the

potential of improvement for motif-aware learning is naturally reduced. In contrast, ProgrammableWeb exhibits richer heterophily where MoHeGCL's motif-type supervised contrast and heterophily-aware propagation are more beneficial.

Looking across cut-offs, Recall increases as K grows, as expected, while NDCG gains are particularly pronounced at smaller and moderate Ks, which indicates better within-top-K ordering. This pattern supports our design: supervising contrast at the motif level improves the discrimination of hard positives and negatives, and the type-aware propagation helps balance feature alignment and feature separation according to local motif composition.

In summary, MoHeGCL delivers consistent and often substantial improvements over all baselines. The gains are most marked on the heterophily-rich ProgrammableWeb dataset, providing empirical evidence that motif-type supervision and heterophily-aware propagation are key to boost top-K recommendation quality.

6.5.4 Ablation Studies

Table 6.2 quantifies the contribution of each component of MoHeGCL by ablating one module at a time on ProgrammableWeb and MovieLens-1M. Across both datasets and all cutoffs, removing any single component degrades performance. Evidently, the features as prior (FP), quantile-scheduled threshold (QST), heterophily-aware propagation (HAP), and soft motif labelling (SML) are all functionally necessary rather than incidental.

QST contributes the largest effect overall, especially on ranking quality. For the ProgrammableWeb dataset, eliminating QST causes the steepest declines. For example, NDCG@20 drops by 9.41% and Recall@10 by 6.52% relative to the full model. Even at broader cutoffs the loss remains substantial, e.g., NDCG@40 drops by 6.62%. MovieLens-1M shows the same pattern with smaller magnitudes, e.g., Recall@10

Table 6.2: Ablation results of MoHeGCL on the ProgrammableWeb and MovieLens-1M datasets. Each row removes one component from the full model. Consistent performance drops confirm the contribution of the features as prior (FP), quantile-scheduled threshold (QST), heterophily-aware propagation (HAP), and soft motif labelling (SML) modules.

Dataset	Variants	Recall@5	NDCG@5	Recall@10	NDCG@10	Recall@20	NDCG@20	Recall@40	NDCG@40
ProgrammableWeb	MoHeGCL	0.4188	0.35571	0.51882	0.39059	0.60763	0.42685	0.69655	0.44023
	– w/o FP	0.3945	0.3382	0.48326	0.368	0.58214	0.39696	0.67229	0.41913
	– w/o QST	0.3922	0.32843	0.485	0.36021	0.57492	0.38667	0.67448	0.41108
	– w/o HAP	0.4006	0.33723	0.51058	0.37534	0.59243	0.40339	0.68555	0.42302
	– w/o SML	0.41019	0.34625	0.50978	0.38088	0.59738	0.40714	0.67889	0.42698
MovieLens-1M	MoHeGCL	0.12093	0.34254	0.19204	0.32289	0.28669	0.32394	0.40929	0.35288
	– w/o FP	0.1202	0.33967	0.18993	0.32	0.2858	0.32196	0.40915	0.35139
	– w/o QST	0.11565	0.33307	0.18476	0.31446	0.27848	0.31553	0.39886	0.3434
	– w/o HAP	0.11956	0.34072	0.18812	0.31889	0.2844	0.32081	0.40411	0.34873
	– w/o SML	0.11946	0.3376	0.18974	0.31941	0.28669	0.32216	0.40846	0.35124

drops by 3.79%, NDCG@40 drops by 2.69%. This aligns with QST's intended role of dynamically calibrating the soft motif decision boundary stabilizes label quality during training, which is particularly beneficial under noisier or more heterophilic interactions as in ProgrammableWeb, and translates directly into better top-rank ordering.

FP delivers the next most significant gains, especially for ProgrammableWeb where Recall@10 declines by 6.85% and NDCG@20 by 7.00% without FP. In contrast, the effect for MovieLens-1M is modest, e.g., Recall@10 drops by 1.10%, NDCG@20 drops by 0.61%. This cross-dataset difference suggests that side-information priors help most when items in triad are more heterophilic and category signals are multi-label and uneven. FP provides a strong inductive bias that improves contrast alignment and propagation in such settings.

HAP yields consistent but moderate improvements, with larger relative gains for ProgrammableWeb, e.g., NDCG@5 drops by 5.20%, NDCG@20 drops by 5.50%, than for MovieLens-1M, e.g., NDCG@10 drops by 1.24%, NDCG@40 drops by 1.18%. The pattern indicates that adapting feature alignment or feature separation by motif type is most valuable in heterophilic settings.

SML contributes smaller yet steady benefits to both Recall and NDCG. For ProgrammableWeb dataset, we observe that NDCG@20 drops by 4.62% and NDCG@40 drops by 3.01% without SML, while for MovieLens-1M shows lighter but consistent drops, e.g., NDCG@5 drops by 1.44%, NDCG@10 drops by 1.08%. This suggests that SML contributes to better ranking refinement especially at moderate cutoffs.

Overall, the improvements persist across K, indicating that each module improves not only hit probability at the very top, i.e. Recall@5 and NDCG@5, but also the broader quality of the ranked list. Overall, the ablation results validate MoHeGCL's design:

- QST is critical for stable motif supervision.
- FP supplies informative priors that are especially beneficial in heterophilic settings.
- HAP tailors propagation to local motif composition.
- SML refines supervision strength to improve rank calibration.

The coherence of these effects across two datasets of different characteristics underscores that the modules are complementary and jointly responsible for the full model's superior performance.

6.6 Conclusion

This chapter introduced MoHeGCL, a motif-based heterophily-aware graph contrastive framework for top-K recommendation. Departing from global homophily assumptions, MoHeGCL treats local triadic motifs as the fundamental supervision unit and adapts both the learning signal and the propagation operator to the motif mix present in each user's neighbourhood. We proposed a soft motif labelling mechanism that assigns motif

types to different types of triads by importing prior features from side information. Our quantile-scheduled thresholding mechanism provides a stable type-adaptive contrastive temperature over training. Through heterophily-aware propagation, MoHeGCL controls feature alignment or separation by the inferred motif proportions via lightweight row scaling.

Empirically, MoHeGCL delivers consistent gains on two datasets with different homophily profiles, outperforming strong LightGCN-based and contrastive baselines across Recall@$\{5, 10, 20, 40\}$ and NDCG@$\{5, 10, 20, 40\}$. Ablation studies show that each component is functionally necessary. QST contributes the largest improvements in ranking quality by stabilizing the supervision signal. FP provides clear benefits when feature priors are informative; HAP further improves robustness personalise propagation strategy for each node in heterophilic neighbourhoods. SML enables smooth transitions between homophilic, mixed, and heterophilic regimes rather than enforcing hard type assignments. Notably, these benefits are obtained with minimal engineering overhead. On top of standard graph contrastive learning architecture, MoHeGCL adds only inexpensive sampling and scaling steps, so the training complexity remains efficient.

While MoHeGCL adapts effectively to mixed levels of homophily, its current motif vocabulary is triad-based and pre-defined. Extending the label space to higher-order structures, learning motifs end-to-end, and incorporating temporal dynamics are promising directions. Finally, transferring MoHeGCL to domains with strong heterophily such as cross-domain recommendation and to non-bipartite settings may broaden its impact.

Overall, this work shows that structure-aware supervision and propagation at the motif level is a general, effective, and efficient approach for recommendation under unknown and shifting homophily levels.

Chapter 7

Conclusion

Across four research components, we show that local graph structure plays a crucial role for recommender systems and that motifs enable GNNs to capture higher-order information that plain GCNs and randomly perturbed contrastive learning methods fail to exploit.

We first introduced MGSR, which learns with motif-level attention on bipartite user–item graphs. By building and weighting multiple motif adjacency matrices, MGSR aggregates higher-order neighbours in a single layer and outperforms strong baselines on the ProgrammableWeb dataset, demonstrating that motifs provide useful signals beyond standard edges.

To make motif usage efficient and scalable, we proposed tailored algorithms to build motif adjacency matrices and introduced MotifGCN with a lightweight propagation rule. MotifGCN reduces complexity compared with attention-based designs while improving recommendation quality on four real-world datasets. Theoretical analysis also explains why motif matrices help mitigate over-squashing compared with standard layer-wise propagation on original adjacency matrix.

We then moved to self-supervision with MGGCL, a motif-guided contrastive learning framework. Instead of applying random augmentations to graph or node features,

we construct two views with intuitive semantics: star-motif views and a rectangle-motif view. Since rectangle edges are contained within the union of star edges, contrasting these views emphasises robust co-consumption patterns and down-weights popularity bias from high-degree nodes, yielding consistent performance improvements on four benchmark datasets and especially on skewed datasets such as Douban-Book and iFashion.

Finally, we addressed heterophily in recommendation by proposing MoHeGCL, which treats each user-anchored triad as a supervision unit. With soft motif labels, a quantile-scheduled threshold and heterophily-aware propagation, MoHeGCL adapts to different homophily levels per user neighbourhood and improves contrastive learning quality while preserving low model complexity.

7.1 Limitations

Despite the strengths of our existing motif-based designs, several limitations remain. First, although the dedicated algorithms for generating motif adjacency matrices are far more efficient than generic subgraph isomorphism procedures, complex motifs such as rectangles can still increase adjacency matrix density and memory consumption on very large graphs, making pre-computation a one-off but non-trivial step. Second, MGSR, MotifGCN and MGGCL rely on a predefined set of seven bipartite motifs, which capture only limited domain semantics, while MoHeGCL focuses exclusively on triads. Third, MGGCL's performance is sensitive to hyperparameters, particularly the contrastive coefficient and the motif ratio that mixes base and motif adjacencies, both of which require dataset-specific tuning. Finally, MGSR and MotifGCN operate on static graphs without considering dynamic and temporal graphs. Although MoHeGCL's soft labels and prior features improve early training, decent side-information are required to fully exploit their potential.

7.2 Future Directions

Future work can proceed in several directions. First, instead of relying on hand-crafted motif sets, one could develop an end-to-end differentiable motif discovery mechanism or neural motif generators that adapt to each domain and extend beyond triads to richer higher-order patterns. Second, the current frameworks could be unified with temporal modelling by extending them to temporal graphs, incorporating time-aware motif counts and triad labels to better handle seasonality and evolving user preferences. Third, scalability remains a key challenge, motivating the exploration of sparse or dynamic motif indices, sketching techniques, and on-the-fly sampling to control adjacency density. These strategies can be combined with quantisation or low-rank adapters to support billion-edge graphs. Fourth, richer supervision signals could be incorporated by generalising MoHeGCL's feature-as-prior design to multi-modal descriptors such as text and images, and by calibrating soft labels with uncertainty estimates to better cope with cold-start users and noisy metadata. Fifth, motif-aware contrastive learning opens opportunities for fairness and bias control, for example by explicitly countering popularity bias through reweighting star versus rectangle signals and by measuring exposure fairness without disproportionately sacrificing accuracy. Finally, there is considerable scope to improve interpretability and diagnostics by developing attribution tools that trace recommendations back to influential motif instances, such as specific rectangles or triads that drive a score. Such tools would turn the structural transparency of MGGCL and MoHeGCL into human-readable explanations.

In conclusion, this thesis establishes a practical strategy for utilising motifs for graph CF by encoding motifs to strengthen message passing, guiding self-supervision with motif semantics, and enabling heterophily-aware propagation. This combination consistently enhances recommender systems while remaining computationally efficient, and it opens a path toward more scalable, interpretable and bias-aware recommender

systems.

References

Abdollahpouri, H., Mansoury, M., Burke, R., Mobasher, B. & Malthouse, E. (2021, June). User-centered Evaluation of Popularity Bias in Recommender Systems. In *Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization* (pp. 119–129). Utrecht Netherlands: ACM. doi: 10.1145/3450613.3456821

Abu-El-Haija, S., Perozzi, B., Kapoor, A., Alipourfard, N., Lerman, K., Harutyunyan, H., ... Galstyan, A. (2019, May). MixHop: Higher-Order Graph Convolutional Architectures via Sparsified Neighborhood Mixing. In *International Conference on Machine Learning* (pp. 21–29). PMLR.

Adomavicius, G. & Tuzhilin, A. (2005, June). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, *17*(6), 734–749. doi: 10.1109/TKDE.2005.99

Alhosaini, H., Alharbi, S., Wang, X. & Xu, G. (2024, May). API Recommendation For Mashup Creation: A Comprehensive Survey. *The Computer Journal*, *67*(5), 1920–1940. doi: 10.1093/comjnl/bxad112

Alon, N., Yuster, R. & Zwick, U. (1995). Color-coding. *Journal of the ACM (JACM)*, *42*(4), 844–856.

Alon, U. (2007, June). Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, *8*(6), 450–461. (Publisher: Nature Publishing Group) doi: 10.1038/nrg2102

Anarfi, R. & Fletcher, K. K. (2019, July). A Reinforcement Learning Approach to Web API Recommendation for Mashup Development. In *2019 IEEE World Congress on Services (SERVICES)* (Vol. 2642-939X, pp. 372–373). (ISSN: 2642-939X) doi: 10.1109/SERVICES.2019.00109

Anwaar, M. U., Han, Z., Arumugaswamy, S., Khan, R. A., Weber, T., Qiu, T., ... Kleinsteuber, M. (2021, April). Metapath and Entity-aware Graph Neural Network for Recommendation. *arXiv:2010.11793 [cs]*.

Babai, L., Erdös, P. & Selkow, S. (1980). Random Graph Isomorphism. *SIAM J. Comput.*. doi: 10.1137/0209047

Bahdanau, D., Cho, K. & Bengio, Y. (2016, May). Neural Machine Translation by Jointly Learning to Align and Translate. In *International Conference on Learning Representations.*

Barabási, A.-L. (2013, March). Network science. *Philosophical Transactions of the*

Royal Society A: Mathematical, Physical and Engineering Sciences, *371*(1987), 20120375. doi: 10.1098/rsta.2012.0375

Bascompte, J. (2009, July). Disentangling the Web of Life. *Science*, *325*(5939), 416–419. (Publisher: American Association for the Advancement of Science) doi: 10.1126/science.1170749

Bastolla, U., Fortuna, M. A., Pascual-García, A., Ferrera, A., Luque, B. & Bascompte, J. (2009, April). The architecture of mutualistic networks minimizes competition and increases biodiversity. *Nature*, *458*(7241), 1018–1020. doi: 10.1038/nature07950

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., … Pascanu, R. (2018, October). Relational inductive biases, deep learning, and graph networks. *arXiv:1806.01261 [cs, stat]*. (arXiv: 1806.01261)

Battaglia, P. W., Pascanu, R., Lai, M., Rezende, D. & Kavukcuoglu, K. (2016, December). *Interaction Networks for Learning about Objects, Relations and Physics*. arXiv. (Issue: arXiv:1612.00222 arXiv: 1612.00222 [cs]) doi: 10.48550/arXiv.1612.00222

Bennett, J. & Lanning, S. (2007, August). The netflix prize. In *Proceedings of the kdd cup 2007*. San Jose, California, USA.

Benson, A. R., Gleich, D. F. & Leskovec, J. (2016, July). Higher-order organization of complex networks. *Science*, *353*(6295), 163–166. doi: 10.1126/science.aad9029

Berg, R. v. d., Kipf, T. N. & Welling, M. (2017, October). Graph Convolutional Matrix Completion. *arXiv:1706.02263 [cs, stat]*.

Billsus, D. & Pazzani, M. J. (2000, June). User Modeling for Adaptive News Access. *User Modeling and User-Adapted Interaction*, *10*(2), 147–180. doi: 10.1023/A:1026501525781

Blei, D. M., Ng, A. Y. & Jordan, M. I. (2003, March). Latent dirichlet allocation. *J. Mach. Learn. Res.*, *3*, 993–1022.

Bo, D., Wang, X., Shi, C. & Shen, H. (2021, May). Beyond Low-frequency Information in Graph Convolutional Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*, 3950–3957. doi: 10.1609/aaai.v35i5.16514

Bodnar, C., Di Giovanni, F., Chamberlain, B., Lió, P. & Bronstein, M. (2022, December). Neural Sheaf Diffusion: A Topological Perspective on Heterophily and Oversmoothing in GNNs. *Advances in Neural Information Processing Systems*, *35*, 18527–18541.

Bouritsas, G., Frasca, F., Zafeiriou, S. P. & Bronstein, M. (2022, February). Improving Graph Neural Network Expressivity via Subgraph Isomorphism Counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*(01), 1–1. doi: 10.1109/TPAMI.2022.3154319

Breese, J. S., Heckerman, D. & Kadie, C. (1998, July). Empirical analysis of predictive algorithms for collaborative filtering. In (pp. 43–52). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A. & Vandergheynst, P. (2017, July). Geometric deep learning: going beyond Euclidean data. *IEEE Signal Processing Magazine*, *34*, 18–42. (arXiv: 1611.08097) doi: 10.1109/MSP.2017.2693418

Bruna, J., Zaremba, W., Szlam, A. & LeCun, Y. (2014, May). Spectral Networks and Locally Connected Networks on Graphs..

Burke, R. (2002, November). Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, *12*(4), 331–370. doi: 10.1023/A:1021240730564

Burke, R. (2007). Hybrid Web Recommender Systems. In P. Brusilovsky, A. Kobsa & W. Nejdl (Eds.), *The Adaptive Web: Methods and Strategies of Web Personalization* (pp. 377–408). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-72079-9_12

Cao, B., Liu, X. F., Rahman, M. M., Li, B., Liu, J. & Tang, M. (2020, January). Integrated Content and Network-Based Service Clustering and Web APIs Recommendation for Mashup Development. *IEEE Transactions on Services Computing*, *13*, 99–113. doi: 10.1109/TSC.2017.2686390

Chanpuriya, S. & Musco, C. (2022, December). Simplified Graph Convolution with Heterophily. *Advances in Neural Information Processing Systems*, *35*, 27184–27197.

Chen, X., Cai, R., Fang, Y., Wu, M., Li, Z. & Hao, Z. (2022, January). Motif Graph Neural Network.

Chen, Z., Villar, S., Chen, L. & Bruna, J. (2019, December). On the equivalence between graph isomorphism testing and function approximation with GNNs. In (pp. 15894–15902). Red Hook, NY, USA: Curran Associates Inc.

Chen, Z., Xu, J., Wei, Y. & Peng, Z. (2025, July). Squeeze and Excitation: A Weighted Graph Contrastive Learning for Collaborative Filtering. In (pp. 2769–2773). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3726302.3730251

Cheng, Z., Walker, J., Zhong, T. & Zhou, F. (2022, July). Modeling Multi-View Interactions with Contrastive Graph Learning for Collaborative Filtering. In (pp. 1–8). doi: 10.1109/IJCNN55064.2022.9892152

Chien, E., Peng, J., Li, P. & Milenkovic, O. (2021, January). Adaptive Universal Generalized PageRank Graph Neural Network. (arXiv: 2006.07988)

Cordella, L., Foggia, P., Sansone, C. & Vento, M. (2004, October). A (sub)graph isomorphism algorithm for matching large graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*, 1367–1372. doi: 10.1109/TPAMI.2004.75

Covington, P., Adams, J. & Sargin, E. (2016, September). Deep Neural Networks for YouTube Recommendations. In (pp. 191–198). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/2959100.2959190

Cui, Z., Cai, Y., Wu, S., Ma, X. & Wang, L. (2021, July). Motif-aware Sequential Recommendation. In (pp. 1738–1742). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3404835.3463115

Dai, J., Li, Q., Nong, T., Bi, Q. & Chu, H. (2024, June). Joint contrastive learning of structural and semantic for graph collaborative filtering. , *586*, 127547. doi: 10.1016/j.neucom.2024.127547

Dareddy, M. R., Das, M. & Yang, H. (2019, August). motif2vec: Motif Aware Node Representation Learning for Heterogeneous Networks.

Das, S., Ghosh, P., Ghosh, S. & Sen, S. (2021, September). Oriented bipartite graphs and the Goldbach graph. *Discrete Mathematics*, *344*(9), 112497. doi: 10.1016/j.disc.2021.112497

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K. & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391–407. (_eprint: https://asistdl.onlinelibrary.wiley.com/doi/pdf/10.1002/%28SICI%291097-4571%28199009%2941%3A6%3C391%3A%3AAID-ASI1%3E3.0.CO%3B2-9) doi: 10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9

Defferrard, M., Bresson, X. & Vandergheynst, P. (2016, December). Convolutional neural networks on graphs with fast localized spectral filtering. In (pp. 3844–3852). Red Hook, NY, USA: Curran Associates Inc.

Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. (2019, June). BERT: Pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran & T. Solorio (Eds.), *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)* (pp. 4171–4186). Minneapolis, Minnesota: Association for Computational Linguistics. doi: 10.18653/v1/N19-1423

Diestel, R. (2017). *Graph Theory* (Fifth Edition ed., Vol. 173). New York: Springer.

Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A. & Adams, R. P. (2015). Convolutional Networks on Graphs for Learning Molecular Fingerprints. In (Vol. 28). Curran Associates, Inc.

Dwivedi, V. P., Luu, A. T., Laurent, T., Bengio, Y. & Bresson, X. (2022, February). *Graph Neural Networks with Learnable Structural and Positional Representations* (Tech. Rep.). arXiv. (arXiv: 2110.07875 [cs]) doi: 10.48550/arXiv.2110.07875

Easley, D., Kleinberg, J. et al. (2010). *Networks, crowds, and markets: Reasoning about a highly connected world* (Vol. 1). Cambridge university press Cambridge.

Fabbri, F., Croci, M. L., Bonchi, F. & Castillo, C. (2022, May). Exposure Inequality in People Recommender Systems: The Long-Term Effects. , *16*, 194–204. doi: 10.1609/icwsm.v16i1.19284

Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J. & Yin, D. (2019, November). Graph Neural Networks for Social Recommendation.

Gao, W., Chen, L., Wu, J. & Gao, H. (2015, June). Manifold-Learning Based API Recommendation for Mashup Creation. In (pp. 432–439). USA: IEEE Computer Society. doi: 10.1109/ICWS.2015.64

Gao, Z., Fan, Y., Wu, C., Tan, W., Zhang, J., Ni, Y., ... Chen, S. (2016, June). SeCo-LDA: Mining Service Co-occurrence Topics for Recommendation. In (pp. 25–32). doi: 10.1109/ICWS.2016.13

Gholinejad, N. & Chehreghani, M. H. (2024, July). *Heterophily-Aware Fair Recommendation using Graph Convolutional Networks.* (Issue: arXiv:2402.03365 arXiv: 2402.03365 [cs])

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O. & Dahl, G. E. (2017, August). Neural message passing for Quantum chemistry. In (pp. 1263–1272). Sydney, NSW, Australia: JMLR.org.

Glorot, X. & Bengio, Y. (2010, March). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (pp. 249–256). JMLR Workshop and Conference Proceedings.

Godsil, C. & Royle, G. F. (2001). *Algebraic Graph Theory* (Vol. 207). Springer Science & Business Media.

Gong, K., Song, X., Li, W. & Wang, S. (2024, January). HN-GCCF: High-order neighbor-enhanced graph convolutional collaborative filtering. , *283*, 111–122. doi: 10.1016/j.knosys.2023.111122

Granovetter, M. S. (1973). The strength of weak ties. *American Journal of Sociology*, *78*(6), 1360–1380.

Hadsell, R., Chopra, S. & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *2006 ieee computer society conference on computer vision and pattern recognition (cvpr'06)* (Vol. 2, p. 1735-1742). doi: 10.1109/CVPR.2006.100

Hamilton, W., Ying, Z. & Leskovec, J. (2017). Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems* (Vol. 30). Curran Associates, Inc.

Hamilton, W. L. (2020). *Graph Representation Learning*.

He, K., Zhang, X., Ren, S. & Sun, J. (2015, December). Deep Residual Learning for Image Recognition.
(arXiv: 1512.03385)

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y. & Wang, M. (2020, July). LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In (pp. 639–648). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3397271.3401063

He, X., Liao, L., Zhang, H., Nie, L., Hu, X. & Chua, T.-S. (2017, August). Neural Collaborative Filtering.

He, X., Zhang, H., Kan, M.-Y. & Chua, T.-S. (2017, August). Fast Matrix Factorization for Online Recommendation with Implicit Feedback.
(arXiv: 1708.05024)

Herlocker, J. L., Konstan, J. A., Borchers, A. & Riedl, J. (1999, August). An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 230–237). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/312624.312682

Herlocker, J. L., Konstan, J. A. & Riedl, J. (2000, December). Explaining collaborative filtering recommendations. In (pp. 241–250). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/358916.358995

Herlocker, J. L., Konstan, J. A., Terveen, L. G. & Riedl, J. T. (2004, January). Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, *22*, 5–53.

doi: 10.1145/963770.963772

Heterophily-Aware Personalized PageRank for Node Classification. (2025, September). In (Vol. 1, pp. 6075–6083). doi: 10.24963/ijcai.2025/676

Hidasi, B. & Karatzoglou, A. (2018, October). Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. In (pp. 843–852). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3269206.3271761

Hidasi, B., Karatzoglou, A., Baltrunas, L. & Tikk, D. (2016, March). *Session-based Recommendations with Recurrent Neural Networks.* (Issue: arXiv:1511.06939 arXiv: 1511.06939 [cs]) doi: 10.48550/arXiv.1511.06939

Holland, P. W. & Leinhardt, S. (1971). Transitivity in structural models of small groups. *Comparative Group Studies*, *2*(2), 107-124. doi: 10.1177/104649647100200201

Hu, Q., Lin, F., Wang, B. & Li, C. (2022, March). MBRep: Motif-based representation learning in heterogeneous networks. *Expert Systems with Applications*, *190*, 116031. doi: 10.1016/j.eswa.2021.116031

Hu, Y., Koren, Y. & Volinsky, C. (2008, December). Collaborative Filtering for Implicit Feedback Datasets. In (pp. 263–272). doi: 10.1109/ICDM.2008.22

Huang, K., Fan, Y. & Tan, W. (2014). Recommendation in an evolving service ecosystem based on network prediction. *IEEE Transactions on Automation Science and Engineering*, *11*(3), 906-920. doi: 10.1109/TASE.2013.2297026

Huang, Q., Xia, X., Xing, Z., Lo, D. & Wang, X. (2018). Api method recommendation without worrying about the task-api knowledge gap. In *Proceedings of the 33rd acm/ieee international conference on automated software engineering* (p. 293–304). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3238147.3238191

Ji, S., Feng, Y., Ji, R., Zhao, X., Tang, W. & Gao, Y. (2020, August). Dual Channel Hypergraph Collaborative Filtering. In (pp. 2020–2029). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3394486.3403253

Jiang, W., Gao, X., Xu, G., Chen, T. & Yin, H. (2024, May). Challenging Low Homophily in Social Recommendation. In (pp. 3476–3484). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3589334.3645460

Jiang, Y., Liu, J., Tang, M. & Liu, X. (2011, July). An Effective Web Service Recommendation Method Based on Personalized Collaborative Filtering. In (pp. 211–218). doi: 10.1109/ICWS.2011.38

Kang, G., Ding, L., Liu, J., Cao, B. & Xu, Y. (2023). Web api recommendation based on self-attentional neural factorization machines with domain interactions. *IEEE Transactions on Network Science and Engineering*, *10*(6), 3953-3963. doi: 10.1109/TNSE.2023.3277695

Kang, G., Liang, B., Liu, J., Wen, Y., Xiao, Y. & Nie, H. (2024, December). Web API Recommendation via Leveraging Content and Network Semantics. *IEEE Transactions on Network and Service Management*, *21*, 5977–5991. doi: 10.1109/TNSM.2024.3422233

Kang, G., Liu, J., Cao, B. & Cao, M. (2020, October). NAFM: Neural and Attentional Factorization Machine for Web API Recommendation. In *2020 IEEE International Conference on Web Services (ICWS)* (pp. 330–337). Beijing, China: IEEE.

doi: 10.1109/ICWS49710.2020.00050

Kang, G., Wang, Y., Ren, H., Cao, B., Liu, J. & Wen, Y. (2024, October). KS-GNN: Keyword Search via Graph Neural Network for Web API Recommendation. *IEEE Trans. on Netw. and Serv. Manag.*, *21*, 5464–5474. doi: 10.1109/TNSM.2024.3420072

Kang, W.-C. & McAuley, J. (2018, November). Self-Attentive Sequential Recommendation. In (pp. 197–206). doi: 10.1109/ICDM.2018.00035

Kashtan, N., Itzkovitz, S., Milo, R. & Alon, U. (2004). Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics*, *20*(11), 1746–1758.

Kingma, D. P. & Ba, J. (2015). Adam: A Method for Stochastic Optimization.. doi: 10.48550/arXiv.1412.6980

Kipf, T., Fetaya, E., Wang, K.-C., Welling, M. & Zemel, R. (2018, July). Neural Relational Inference for Interacting Systems. In (pp. 2688–2697). PMLR.

Kipf, T. N. & Welling, M. (2017, February). Semi-Supervised Classification with Graph Convolutional Networks..

Koren, Y. (2008, August). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In (pp. 426–434). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/1401890.1401944

Koren, Y., Bell, R. & Volinsky, C. (2009, August). Matrix Factorization Techniques for Recommender Systems. *Computer*, *42*, 30–37. doi: 10.1109/MC.2009.263

Kreuzer, D., Beaini, D., Hamilton, W. L., Létourneau, V. & Tossou, P. (2021, May). Rethinking Graph Transformers with Spectral Attention. In *Advances in Neural Information Processing Systems.*

Lam, X. N., Vu, T., Le, T. D. & Duong, A. D. (2008). Addressing cold-start problem in recommendation systems. In *Proceedings of the 2nd international conference on ubiquitous information management and communication* (p. 208–211). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/1352793.1352837

Le, Q. & Mikolov, T. (2014, 22–24 Jun). Distributed representations of sentences and documents. In E. P. Xing & T. Jebara (Eds.), *Proceedings of the 31st international conference on machine learning* (Vol. 32, pp. 1188–1196). Bejing, China: PMLR.

Lee, D. & Seung, H. S. (2000). Algorithms for Non-negative Matrix Factorization. In T. Leen, T. Dietterich & V. Tresp (Eds.), *Advances in Neural Information Processing Systems* (Vol. 13). MIT Press.

Lee, J. B., Rossi, R. A., Kong, X., Kim, S., Koh, E. & Rao, A. (2019, November). Graph Convolutional Networks with Motif-based Attention. In (pp. 499–508). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3357384.3357880

Levie, R., Monti, F., Bresson, X. & Bronstein, M. M. (2019, January). CayleyNets: Graph Convolutional Neural Networks With Complex Rational Spectral Filters. *IEEE Transactions on Signal Processing*, *67*, 97–109. doi: 10.1109/TSP.2018.2879624

Li, C., Guo, Y., Tang, Z., Li, W., Chen, Y., Cao, J., … Jiang, J. (2025, August). MHGCL:

Combining Motif and Homogeneity in graph contrastive learning. *World Wide Web*, *28*, 1–22. doi: 10.1007/s11280-025-01361-z

Li, H., Liu, J., Cao, B., Tang, M., Liu, X. & Li, B. (2017, June). Integrating Tag, Topic, Co-Occurrence, and Popularity to Recommend Web APIs for Mashup Creation. In (pp. 84–91). (ISSN: 2474-2473) doi: 10.1109/SCC.2017.19

Li, H., Wang, B., Cui, L., Bai, L. & Hancock, E. (2020). LGL-GNN: Learning Global and Local Information for Graph Neural Networks.. doi: 10.1007/978-3-030-73973-7_13

Li, R., Wang, S., Zhu, F. & Huang, J. (2018, April). Adaptive Graph Convolutional Neural Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 32).

Li, X., Tian, Y., Dong, B. & Ji, S. (2024, July). MD-GCCF: Multi-view deep graph contrastive learning for collaborative filtering. , *590*, 127756. doi: 10.1016/j.neucom.2024.127756

Li, X., Wei, W., Feng, X., Liu, X. & Zheng, Z. (2021, November). Representation learning of graphs using graph convolutional multilayer networks based on motifs. *Neurocomputing*, *464*, 218–226. doi: 10.1016/j.neucom.2021.08.028

Li, X., Zhu, R., Cheng, Y., Shan, C., Luo, S., Li, D. & Qian, W. (2022, June). Finding Global Homophily in Graph Neural Networks When Meeting Heterophily. In (pp. 13242–13256). PMLR.

Li, Y., Tarlow, D., Brockschmidt, M. & Zemel, R. (2017, September). Gated Graph Sequence Neural Networks..

Li, Y., Yu, R., Shahabi, C. & Liu, Y. (2018, February). *Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting.* (Issue: arXiv:1707.01926 arXiv: 1707.01926 [cs]) doi: 10.48550/arXiv.1707.01926

Lian, S. & Tang, M. (2022, December). API recommendation for Mashup creation based on neural graph collaborative filtering. *Connection Science*, *34*, 124–138. (Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/09540091.2021.1974819) doi: 10.1080/09540091.2021.1974819

Liang, H., Du, X., Zhu, B., Ma, Z., Chen, K. & Gao, J. (2023, June). Graph contrastive learning with implicit augmentations. *Neural Networks*, *163*, 156–164. doi: 10.1016/j.neunet.2023.04.001

Liang, T., Chen, L., Wu, J., Dong, H. & Bouguettaya, A. (2016). Meta-Path Based Service Recommendation in Heterogeneous Information Networks. In Q. Z. Sheng, E. Stroulia, S. Tata & S. Bhiri (Eds.), *Service-Oriented Computing* (pp. 371–386). Cham: Springer International Publishing. doi: 10.1007/978-3-319-46295-0_23

LightGCL: Simple Yet Effective Graph Contrastive Learning for Recommendation. (2022, September).

Lima, E. & Liu, X. (2025, April). Learning optimal heterogeneous service network representation. *Service Oriented Computing and Applications*. doi: 10.1007/s11761-025-00453-y

Lin, W., Xiao, X., Xie, X. & Li, X.-L. (2016). Network motif discovery: A gpu approach. *IEEE transactions on knowledge and data engineering*, *29*(3), 513–528.

Lin, Z., Tian, C., Hou, Y. & Zhao, W. X. (2022, April). Improving Graph Collaborative Filtering with Neighborhood-enriched Contrastive Learning. In (pp. 2320–2329). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3485447.3512104

Linden, G., Smith, B. & York, J. (2003). Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, *7*(1), 76-80. doi: 10.1109/MIC.2003.1167344

Liu, S., Yao, S., Huang, Y., Liu, D., Shao, H., Zhao, Y., ... Abdelzaher, T. (2020, September). Handling Missing Sensors in Topology-Aware IoT Applications with Gated Graph Neural Network. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, *4*, 90:1–90:31. doi: 10.1145/3411818

Liu, X., Hao, Y., Zhao, L., Liu, G., Sheng, V. S. & Zhao, P. (2024, August). LMACL: Improving Graph Collaborative Filtering with Learnable Model Augmentation Contrastive Learning. *ACM Transactions on Knowledge Discovery from Data*, *18*, 1–24. doi: 10.1145/3657302

Liu, X. & Song, Y. (2022, February). Graph Convolutional Networks with Dual Message Passing for Subgraph Isomorphism Counting and Matching. In *36th AAAI Conference On Artificial Intelligence*.

Liu, X., Wen, G., Wang, A., Liu, C., Wei, W. & Meo, P. D. (2025). GGDHSCL: A Graph Generative Diffusion With Hard Negative Sampling Contrastive Learning Recommendation Method. , 1–16. doi: 10.1109/TCSS.2025.3539854

Liu, Y., Jin, M., Pan, S., Zhou, C., Zheng, Y., Xia, F. & Yu, P. S. (2023, June). Graph Self-Supervised Learning: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, *35*, 5879–5900. doi: 10.1109/TKDE.2022.3172903

Liu, Y., Kertkeidkachorn, N., Miyazaki, J. & Ichise, R. (2025, June). Review-enhanced contrastive learning on knowledge graphs for recommendation. *Expert Systems with Applications*, *277*, 127250. doi: 10.1016/j.eswa.2025.127250

Liu, Y., Schmidt, B., Liu, W. & Maskell, D. L. (2010). Cuda–meme: accelerating motif discovery in biological sequences using cuda-enabled graphics processing units. *Pattern Recognition Letters*, *31*(14), 2170–2177.

Liu, Y., Zheng, Y., Zhang, D., Lee, V. C. & Pan, S. (2023, June). Beyond Smoothing: Unsupervised Graph Representation Learning with Edge Heterophily Discriminating. *Proceedings of the AAAI Conference on Artificial Intelligence*, *37*, 4516–4524. doi: 10.1609/aaai.v37i4.25573

Lops, P., de Gemmis, M. & Semeraro, G. (2011). Content-based Recommender Systems: State of the Art and Trends. In F. Ricci, L. Rokach, B. Shapira & P. B. Kantor (Eds.), (pp. 73–105). Boston, MA: Springer US. doi: 10.1007/978-0-387-85820-3_3

Luan, S., Hua, C., Lu, Q., Zhu, J., Zhao, M., Zhang, S., ... Precup, D. (2022, October). *Revisiting Heterophily For Graph Neural Networks.* (Issue: arXiv:2210.07606 arXiv: 2210.07606 [cs]) doi: 10.48550/arXiv.2210.07606

Ludewig, M. & Jannach, D. (2018, December). Evaluation of session-based recommendation algorithms. *User Modeling and User-Adapted Interaction*, *28*, 331–390. doi: 10.1007/s11257-018-9209-6

Lü, L., Medo, M., Yeung, C. H., Zhang, Y.-C., Zhang, Z.-K. & Zhou, T. (2012, October). Recommender systems. *Physics Reports*, *519*, 1–49. doi: 10.1016/j.physrep.2012.02.006

Ma, Y., Geng, X. & Wang, J. (2021, February). A Deep Neural Network With Multiplex Interactions for Cold-Start Service Recommendation. *IEEE Transactions on Engineering Management*, *68*, 105–119. doi: 10.1109/TEM.2019.2961376

Ma, Y., Liu, X., Shah, N. & Tang, J. (2023, July). *Is Homophily a Necessity for Graph Neural Networks?* (Issue: arXiv:2106.06134 arXiv: 2106.06134 [cs]) doi: 10.48550/arXiv.2106.06134

Mao, K., Zhu, J., Wang, J., Dai, Q., Dong, Z., Xiao, X. & He, X. (2021, October). SimpleX: A Simple and Strong Baseline for Collaborative Filtering. In (pp. 1243–1252). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3459637.3482297

Maron, H., Ben-Hamu, H., Serviansky, H. & Lipman, Y. (2019, December). Provably Powerful Graph Networks. In (pp. 2156–2167). Red Hook, NY, USA: Curran Associates Inc.

Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013). *Efficient estimation of word representations in vector space.*

Milo, R., Itzkovitz, S., Kashtan, N., Levitt, R., Shen-Orr, S., Ayzenshtat, I., ... Alon, U. (2004, March). Superfamilies of Evolved and Designed Networks. *Science*, *303*, 1538–1542. (Publisher: American Association for the Advancement of Science) doi: 10.1126/science.1089167

Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D. & Alon, U. (2002, October). Network Motifs: Simple Building Blocks of Complex Networks. *Science*, *298*(5594), 824–827. doi: 10.1126/science.298.5594.824

Mnih, A. & Salakhutdinov, R. R. (2008). Probabilistic Matrix Factorization. In J. C. Platt, D. Koller, Y. Singer & S. T. Roweis (Eds.), *Advances in Neural Information Processing Systems 20* (pp. 1257–1264). Curran Associates, Inc.

Monti, F., Bronstein, M. M. & Bresson, X. (2017, April). Geometric Matrix Completion with Recurrent Multi-Graph Neural Networks.

Monti, F., Otness, K. & Bronstein, M. M. (2018, June). MOTIFNET: A MOTIF-BASED GRAPH CONVOLUTIONAL NETWORK FOR DIRECTED GRAPHS. In (pp. 225–228). doi: 10.1109/DSW.2018.8439897

Mooney, R. J. & Roy, L. (2000, June). Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries* (pp. 195–204). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/336597.336662

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G. & Grohe, M. (2019, January). Weisfeiler and leman go neural: higher-order graph neural networks. In (pp. 4602–4609). Honolulu, Hawaii, USA: AAAI Press. doi: 10.1609/aaai.v33i01.33014602

Newman, M. (2018). *Networks*. doi: 10.1093/oso/9780198805090.001.0001

Newman, M. E. J. (2003). The structure and function of complex networks. *SIAM Review*, *45*(2), 167–256. Retrieved from `https://doi.org/10.1137/`

S003614450342480 doi: 10.1137/S003614450342480

Nguyen, M., Yu, J., Nguyen, T. & Han, Y. (2021, December). Attentional matrix factorization with context and co-invocation for service recommendation. *Expert Systems with Applications*, *186*, 115698. doi: 10.1016/j.eswa.2021.115698

Nguyen, M., Yu, J., Nguyen, T. & Yongchareon, S. (2022, March). High-order autoencoder with data augmentation for collaborative filtering. *Knowledge-Based Systems*, *240*, 107773. doi: 10.1016/j.knosys.2021.107773

Ni, X., Xiong, F., Zheng, Y. & Wang, L. (2024, May). Graph Contrastive Learning with Kernel Dependence Maximization for Social Recommendation. In *Proceedings of the ACM Web Conference 2024* (pp. 481–492). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3589334.3645412

Oord, A. v. d., Li, Y. & Vinyals, O. (2019, January). *Representation Learning with Contrastive Predictive Coding.* (Issue: arXiv:1807.03748 arXiv: 1807.03748 [cs]) doi: 10.48550/arXiv.1807.03748

Ouyang, Z., Zhang, C., Hou, S., Zhang, C. & Ye, Y. (2024, May). How to Improve Representation Alignment and Uniformity in Graph-Based Collaborative Filtering? In (Vol. 18, pp. 1148–1159). doi: 10.1609/icwsm.v18i1.31379

Pan, S. J. & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345-1359. doi: 10.1109/TKDE.2009.191

Pan, Z. & Chen, H. (2021, January). Efficient Graph Collaborative Filtering via Contrastive Learning. *Sensors*, *21*, 4666. doi: 10.3390/s21144666

Pazzani, M. & Billsus, D. (1997, June). Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Machine Learning*, *27*(3), 313–331. doi: 10.1023/A:1007369909943

Pazzani, M. J. & Billsus, D. (2007). Content-Based Recommendation Systems. In P. Brusilovsky, A. Kobsa & W. Nejdl (Eds.), *The Adaptive Web: Methods and Strategies of Web Personalization* (pp. 325–341). Berlin, Heidelberg: Springer. doi: 10.1007/978-3-540-72079-9_10

Pei, H., Wei, B., Chang, K. C.-C., Lei, Y. & Yang, B. (2019, September). Geom-GCN: Geometric Graph Convolutional Networks. In *International Conference on Learning Representations.*

Peng, H., Li, J., Gong, Q., Ning, Y., Wang, S. & He, L. (2020, April). Motif-Matching Based Subgraph-Level Attentional Convolutional Network for Graph Classification. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, pp. 5387–5394). doi: 10.1609/aaai.v34i04.5987

Peng, S., Sugiyama, K. & Mine, T. (2022, July). Less is More: Reweighting Important Spectral Graph Features for Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 1273–1282). Madrid Spain: ACM. doi: 10.1145/3477495.3532014

Pennington, J., Socher, R. & Manning, C. (2014, October). GloVe: Global Vectors for Word Representation. In A. Moschitti, B. Pang & W. Daelemans (Eds.), *Proceedings of the 2014 Conference on Empirical Methods in Natural Language*

Processing (EMNLP) (pp. 1532–1543). Doha, Qatar: Association for Computational Linguistics. doi: 10.3115/v1/D14-1162

Piao, J., Zhang, G., Xu, F., Chen, Z. & Li, Y. (2021, April). Predicting Customer Value with Social Relationships via Motif-based Graph Attention Networks. In (pp. 3146–3157). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3442381.3449849

ProgrammableWeb has been retired. (n.d.).

Protogerou, A., Papadopoulos, S., Drosou, A., Tzovaras, D. & Refanidis, I. (2021, March). A graph neural network method for distributed anomaly detection in IoT. *Evolving Systems*, *12*(1), 19–36. doi: 10.1007/s12530-020-09347-0

Qi, L., He, Q., Chen, F., Dou, W., Wan, S., Zhang, X. & Xu, X. (2019). Finding all you need: Web apis recommendation in web of things through keywords search. *IEEE Transactions on Computational Social Systems*, *6*(5), 1063-1072. doi: 10.1109/TCSS.2019.2906925

Qiu, C., Nan, G., Xiong, T., Deng, W., Wang, D., Teng, Z., ... Tao, X. (2024, March). Refining Latent Homophilic Structures over Heterophilic Graphs for Robust Graph Convolution Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, *38*, 8930–8938. doi: 10.1609/aaai.v38i8.28741

Qiu, J., Chen, Q., Dong, Y., Zhang, J., Yang, H., Ding, M., ... Tang, J. (2020, August). GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. In (pp. 1150–1160). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3394486.3403168

Quadrana, M., Cremonesi, P. & Jannach, D. (2018, July). Sequence-Aware Recommender Systems. *ACM Comput. Surv.*, *51*, 66:1–66:36. doi: 10.1145/3190616

Rahman, M., Bhuiyan, M. A. & Al Hasan, M. (2014). Graft: An efficient graphlet counting method for large graph analysis. *IEEE Transactions on Knowledge and Data Engineering*, *26*(10), 2466-2478. doi: 10.1109/TKDE.2013.2297929

Rahman, M. M., Liu, X. & Cao, B. (2017, June). Web API Recommendation for Mashup Development Using Matrix Factorization on Integrated Content and Network-Based Service Clustering. In (pp. 225–232). (ISSN: 2474-2473) doi: 10.1109/SCC.2017.36

Rahman, M. M. & Liu, X. F. (2020, November). Integrated Topic Modeling and User Interaction Enhanced WebAPI Recommendation using Regularized Matrix Factorization for Mashup Application Development. In (pp. 124–131). (ISSN: 2474-2473) doi: 10.1109/SCC49832.2020.00025

Rendle, S. (2010, December). Factorization Machines. In (pp. 995–1000). doi: 10.1109/ICDM.2010.127

Rendle, S., Freudenthaler, C., Gantner, Z. & Schmidt-Thieme, L. (2009). BPR: Bayesian personalized ranking from implicit feedback. In (pp. 452–461). Arlington, Virginia, USA: AUAI Press.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. & Riedl, J. (1994, October). GroupLens: an open architecture for collaborative filtering of netnews. In (pp. 175–186). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/192844.192905

Ricci, F., Rokach, L. & Shapira, B. (2011). Introduction to Recommender Systems Handbook. In F. Ricci, L. Rokach, B. Shapira & P. B. Kantor (Eds.), (pp. 1–35). Boston, MA: Springer US. doi: 10.1007/978-0-387-85820-3_1

Ricci, F., Rokach, L. & Shapira, B. (2015). Recommender Systems: Introduction and Challenges. In F. Ricci, L. Rokach & B. Shapira (Eds.), *Recommender Systems Handbook* (pp. 1–34). Boston, MA: Springer US. doi: 10.1007/978-1-4899-7637-6_1

Rosas-Casals, M., Valverde, S. & Solé, R. V. (2007, July). Topological vulnerability of the european power grid under errors and attacks. *International Journal of Bifurcation and Chaos*, *17*, 2465–2475. (Publisher: World Scientific Publishing Co.) doi: 10.1142/S0218127407018531

Rossi, E., Charpentier, B., Di Giovanni, F., Frasca, F., Günnemann, S. & Bronstein, M. (2023, November). *Edge Directionality Improves Learning on Heterophilic Graphs.* (Issue: arXiv:2305.10498 arXiv: 2305.10498 [cs]) doi: 10.48550/arXiv.2305.10498

Rossi, R. A., Ahmed, N. K., Carranza, A., Arbour, D., Rao, A., Kim, S. & Koh, E. (2019). Heterogeneous network motifs. *arXiv preprint arXiv:1901.10026*.

Rossi, R. A., Ahmed, N. K. & Koh, E. (2018). Higher-order Network Representation Learning. In *Companion of the The Web Conference 2018 on The Web Conference 2018 - WWW '18* (pp. 3–4). Lyon, France: ACM Press. doi: 10.1145/3184558.3186900

Salakhutdinov, R. & Mnih, A. (2007, December). Probabilistic Matrix Factorization. In (pp. 1257–1264). Red Hook, NY, USA: Curran Associates Inc.

Salton, G. & Buckley, C. (1988, January). Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, *24*, 513–523. doi: 10.1016/0306-4573(88)90021-0

Salton, G., Wong, A. & Yang, C. S. (1975, November). A vector space model for automatic indexing. *Commun. ACM*, *18*, 613–620. doi: 10.1145/361219.361220

Sang, C., Deng, X. & Liao, S. (2023, January). Mashup-Oriented Web API Recommendation Via Full-Text Semantic Mining of Developer Requirements. , *16*, 2755–2768. (Publisher: IEEE ADS Bibcode: 2023ITSCo..16.2755S) doi: 10.1109/TSC.2023.3245652

Sankar, A., Wang, J., Krishnan, A. & Sundaram, H. (2020, November). Beyond Localized Graph Neural Networks: An Attributed Motif Regularization Framework. In (pp. 472–481). doi: 10.1109/ICDM50108.2020.00056

Sankar, A., Zhang, X. & Chang, K. C.-C. (2019, July). Motif-based Convolutional Neural Network on Graphs.

Sarwar, B., Karypis, G., Konstan, J. & Riedl, J. (2001, April). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web* (pp. 285–295). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/371920.372071

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M. & Monfardini, G. (2009, January). The Graph Neural Network Model. *IEEE Transactions on Neural Networks*, *20*, 61–80. doi: 10.1109/TNN.2008.2005605

Schlichtkrull, M., Kipf, T. N., Bloem, P., Berg, R. v. d., Titov, I. & Welling, M. (2017, October). *Modeling Relational Data with Graph Convolutional Networks.* (Issue: arXiv:1703.06103 arXiv: 1703.06103 [stat]) doi: 10.48550/arXiv.1703.06103

Sedhain, S., Menon, A. K., Sanner, S. & Xie, L. (2015). Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th international conference on world wide web* (p. 111–112). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/2740908.2742726

Shani, G., Brafman, R. I. & Heckerman, D. (2002, August). An MDP-based recommender system. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence* (pp. 453–460). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Shen, L., Wang, Y., Zhang, S. & Chen, Z. (2023, January). Similarity and Complementarity Attention-Based Graph Neural Networks for Mashup-Oriented Cloud API Recommendation. *Electronics*, *12*(21), 4436. (Publisher: Multidisciplinary Digital Publishing Institute) doi: 10.3390/electronics12214436

Sheng, J., Zhang, Y., Wang, B. & Chang, Y. (2024, January). MGATs: Motif-Based Graph Attention Networks. *Mathematics*, *12*, 293. doi: 10.3390/math12020293

Shen-Orr, S. S., Milo, R., Mangan, S. & Alon, U. (2002, May). Network motifs in the transcriptional regulation network of Escherichia coli. *Nature Genetics*, *31*(1), 64–68. (Publisher: Nature Publishing Group) doi: 10.1038/ng881

Shervashidze, N., Vishwanathan, S., Petri, T., Mehlhorn, K. & Borgwardt, K. (2009). Efficient graphlet kernels for large graph comparison. In *Artificial intelligence and statistics* (pp. 488–495).

Shi, T., Si, Z., Xu, J., Zhang, X., Zang, X., Zheng, K., … Song, Y. (2024, July). UniSAR: Modeling User Transition Behaviors between Search and Recommendation. In (pp. 1029–1039). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3626772.3657811

Soundarajan, S., Eliassi-Rad, T. & Gallagher, B. (2014). A guide to selecting a network similarity method. In *Proceedings of the 2014 siam international conference on data mining* (pp. 1037–1045).

Stone, L., Simberloff, D. & Artzy-Randrup, Y. (2019, April). Network motifs and their origins. *PLOS Computational Biology*, *15*, e1006749. doi: 10.1371/journal.pcbi.1006749

Stouffer, D. B., Camacho, J., Jiang, W. & Nunes Amaral, L. A. (2007, August). Evidence for the existence of a robust pattern of prey selection in food webs. *Proceedings of the Royal Society B: Biological Sciences*, *274*, 1931–1940. doi: 10.1098/rspb.2007.0571

Su, X. & Khoshgoftaar, T. M. (2009). A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence*, *2009*(1), 421425. (_eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1155/2009/421425) doi: 10.1155/2009/421425

Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W. & Jiang, P. (2019, November). BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In (pp. 1441–1450). New York, NY, USA: Association

for Computing Machinery. doi: 10.1145/3357384.3357895

Sun, Y., Zhu, D., Du, H. & Tian, Z. (2022, November). Motifs-based recommender system via hypergraph convolution and contrastive learning. , *512*, 323–338. doi: 10.1016/j.neucom.2022.09.102

Sun, Z., Yu, D., Fang, H., Yang, J., Qu, X., Zhang, J. & Geng, C. (2020, September). Are We Evaluating Rigorously? Benchmarking Recommendation for Reproducible Evaluation and Fair Comparison. In (pp. 23–32). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3383313.3412489

Tan, P.-N., Steinbach, M. & Kumar, V. (2016). *Introduction to data mining*. Pearson Education India.

Tang, J. & Wang, K. (2018, February). Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In (pp. 565–573). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3159652.3159656

Tang, M., Xie, F., Lian, S., Mai, J. & Li, S. (2024, May). Mashup-oriented API recommendation via pre-trained heterogeneous information networks. , *169*, 107428. doi: 10.1016/j.infsof.2024.107428

Trivedi, P., Lubana, E. S., Yan, Y., Yang, Y. & Koutra, D. (2022, April). Augmentations in Graph Contrastive Learning: Current Methodological Flaws & Towards Better Practices. In *Proceedings of the ACM Web Conference 2022* (pp. 1538–1549). Virtual Event, Lyon France: ACM. doi: 10.1145/3485447.3512200

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., … Polosukhin, I. (2017). Attention is All you Need. In *Advances in Neural Information Processing Systems* (Vol. 30). Curran Associates, Inc.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P. & Bengio, Y. (2018, February). Graph Attention Networks. In *International Conference on Learning Representations.*

Veličković, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y. & Hjelm, R. D. (2019). Deep Graph Infomax. In *International Conference on Learning Representations.*

Wang, G., Yu, J., Nguyen, M., Zhang, Y., Yongchareon, S. & Han, Y. (2023, June). Motif-based graph attentional neural network for web service recommendation. , *269*, 110512. doi: 10.1016/j.knosys.2023.110512

Wang, H., Solin, A. & Garg, V. K. (2025, January). Heterophily-informed Message Passing.

Wang, H., Zhang, F., Zhang, M., Leskovec, J., Zhao, M., Li, W. & Wang, Z. (2019, June). Knowledge-aware Graph Neural Networks with Label Smoothness Regularization for Recommender Systems.

Wang, J., Ding, K., Hong, L., Liu, H. & Caverlee, J. (2020, July). Next-item Recommendation with Sequential Hypergraphs. In (pp. 1101–1110). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3397271.3401133

Wang, J., Guo, Y., Yang, L. & Wang, Y. (2024, July). Understanding Heterophily for Graph Neural Networks. In (pp. 50489–50529). PMLR.

Wang, J., Zhang, S., Xiao, Y. & Song, R. (2022, April). *A Review on Graph Neural Network Methods in Financial Applications.* (Issue: arXiv:2111.15367 arXiv: 2111.15367 [q-fin]) doi: 10.48550/arXiv.2111.15367

Wang, S., Zhou, Y., Fan, X., Li, J., Lei, Z. & Gong, M. (2025). Personalized Federated Contrastive Learning for Recommendation. , 1–13. doi: 10.1109/TCSS.2024.3524587

Wang, X., He, X., Cao, Y., Liu, M. & Chua, T.-S. (2019, July). KGAT: Knowledge Graph Attention Network for Recommendation. In (pp. 950–958). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3292500.3330989

Wang, X., He, X., Wang, M., Feng, F. & Chua, T.-S. (2019, July). Neural Graph Collaborative Filtering. In (pp. 165–174). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3331184.3331267

Wang, X., Liu, X., Liu, J., Chen, X. & Wu, H. (2021, May). A novel knowledge graph embedding based API recommendation method for Mashup development. *World Wide Web*, *24*(3), 869–894. doi: 10.1007/s11280-021-00894-3

Wang, X., Wu, H. & Hsu, C.-H. (2019). Mashup-Oriented API Recommendation via Random Walk on Knowledge Graph. , *7*, 7651–7662. doi: 10.1109/ACCESS.2018.2890156

Wang, Y., Chen, J., Huang, Q., Xia, X. & Jiang, B. (2023, June). Deep learning-based open API recommendation for Mashup development. *Science China Information Sciences*, *66*(7), 172102. doi: 10.1007/s11432-021-3531-0

Wang, Z., Wei, W., Cong, G., Li, X.-L., Mao, X.-L. & Qiu, M. (2020, July). Global Context Enhanced Graph Neural Networks for Session-based Recommendation. In (pp. 169–178). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3397271.3401142

Wasserman, S. & Faust, K. (1994). *Social network analysis: Methods and applications*. Cambridge University Press.

Wei, C., Wang, Y., Bai, B., Ni, K., Brady, D. & Fang, L. (2023, July). Boosting Graph Contrastive Learning via Graph Contrastive Saliency. In *Proceedings of the 40th International Conference on Machine Learning* (pp. 36839–36855). PMLR.

Wei, H., Wang, J., Ji, Y., Guang, M. & Yan, C. (2025, April). Hierarchical neighbor-enhanced graph contrastive learning for recommendation. , *315*, 113263. doi: 10.1016/j.knosys.2025.113263

Weinberger, K. Q. & Saul, L. K. (2009). Distance Metric Learning for Large Margin Nearest Neighbor Classification. *Journal of Machine Learning Research*, *10*(9), 207–244.

Weisfeiler, B. & Leman, A. (1968). The reduction of a graph to canonical form and the algebra which appears therein. *Nauchno-Technicheskaya Informatsia*, *2*(9), 12–16.

Wernicke, S. & Rasche, F. (2006, 02). Fanmod: a tool for fast network motif detection. *Bioinformatics*, *22*(9), 1152-1153. doi: 10.1093/bioinformatics/btl038

West, D. B. & others. (2001). *Introduction to graph theory* (Vol. 2). Prentice hall Upper Saddle River.

Wu, C., Wu, F., Ge, S., Qi, T., Huang, Y. & Xie, X. (2019, November). Neural News Recommendation with Multi-Head Self-Attention. In K. Inui, J. Jiang, V. Ng & X. Wan (Eds.), (pp. 6389–6394). Hong Kong, China: Association for Computational Linguistics. doi: 10.18653/v1/D19-1671

Wu, F., Zhang, T., Souza Jr., A. H. d., Fifty, C., Yu, T. & Weinberger, K. Q. (2019, June). Simplifying Graph Convolutional Networks.

Wu, H., Duan, Y., Yue, K. & Zhang, L. (2022, November). Mashup-Oriented Web API Recommendation via Multi-Model Fusion and Multi-Task Learning. *IEEE Transactions on Services Computing*, *15*, 3330–3343. doi: 10.1109/TSC.2021.3098756

Wu, J., Wang, X., Feng, F., He, X., Chen, L., Lian, J. & Xie, X. (2021, July). Self-supervised Graph Learning for Recommendation. In (pp. 726–735). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3404835.3462862

Wu, L., He, X., Wang, X., Zhang, K. & Wang, M. (2023, May). A Survey on Accuracy-Oriented Neural Recommendation: From Collaborative Filtering to Information-Rich Recommendation. *IEEE Transactions on Knowledge and Data Engineering*, *35*, 4425–4445. doi: 10.1109/TKDE.2022.3145690

Wu, Q., Zhang, H., Gao, X., He, P., Weng, P., Gao, H. & Chen, G. (2019, May). Dual Graph Attention Networks for Deep Latent Representation of Multifaceted Social Effects in Recommender Systems. In (pp. 2091–2102). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3308558.3313442

Wu, X., Wang, C.-D., Lin, J.-Q., Xi, W.-D. & Yu, P. S. (2024, September). Motif-Based Contrastive Learning for Community Detection. *IEEE Transactions on Neural Networks and Learning Systems*, *35*, 11706–11719. doi: 10.1109/TNNLS.2024.3367873

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C. & Yu, P. S. (2021, January). A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, *32*, 4–24. doi: 10.1109/TNNLS.2020.2978386

Xia, B., Fan, Y., Tan, W., Huang, K., Zhang, J. & Wu, C. (2015, September). Category-Aware API Clustering and Distributed Recommendation for Automatic Mashup Creation. *IEEE Transactions on Services Computing*, *8*, 674–687. doi: 10.1109/TSC.2014.2379251

Xia, J., Li, D., Gu, H., Lu, T., Zhang, P., Shang, L. & Gu, N. (2024, May). Hierarchical Graph Signal Processing for Collaborative Filtering. In (pp. 3229–3240). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3589334.3645368

Xia, L., Huang, C., Xu, Y., Zhao, J., Yin, D. & Huang, J. (2022, July). Hypergraph Contrastive Collaborative Filtering. In (pp. 70–79). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3477495.3532058

Xie, F., Chen, L., Lin, D., Zheng, Z. & Lin, X. (2019). Personalized Service Recommendation With Mashup Group Preference in Heterogeneous Information Network. , *7*, 16155–16167. doi: 10.1109/ACCESS.2019.2894822

Xie, F., Wang, J., Xiong, R., Zhang, N., Ma, Y. & He, K. (2019, June). An integrated service recommendation approach for service-based system development. *Expert Systems with Applications*, *123*, 178–194. doi: 10.1016/j.eswa.2019.01.025

Xu, B., Shen, H., Cao, Q., Qiu, Y. & Cheng, X. (2018, September). Graph Wavelet Neural Network. In *International Conference on Learning Representations.*

Xu, J., Chen, Z., Yang, S., Li, J., Wang, H., Wang, W., … Ngai, E. (2025). NLGCL: Naturally Existing Neighbor Layers Graph Contrastive Learning for Recommendation. Prague, Czech Republic. doi: 10.48550/arXiv.2507.07522

Xu, K., Hu, W., Leskovec, J. & Jegelka, S. (2019). How Powerful are Graph Neural Networks? In *International Conference on Learning Representations.*

Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.-i. & Jegelka, S. (2018, June). Representation Learning on Graphs with Jumping Knowledge Networks.

Xu, W., Cao, J., Hu, L., Wang, J. & Li, M. (2013, June). A Social-Aware Service Recommendation Approach for Mashup Creation. In (pp. 107–114). doi: 10.1109/ICWS.2013.24

Xu, W., Gao, F., Su, J., Wu, Q. & Wang, M. (2024). Dual-view Enhanced Knowledge Contrastive Learning for Recommendation. In (pp. 528–538). Springer, Singapore. doi: 10.1007/978-981-97-5562-2_36

Xue, J., Yang, Z., Lin, H., Zhang, Z., Wang, L., Gu, Y., … Li, X. (2025, May). *HGCL: Hierarchical Graph Contrastive Learning for User-Item Recommendation.* arXiv. (Issue: arXiv:2505.19020 arXiv: 2505.19020 [cs]) doi: 10.48550/arXiv.2505.19020

Xue, Q., Liu, L., Chen, W. & Chuah, M. C. (2017, December). Automatic Generation and Recommendation for API Mashups. In (pp. 119–124). doi: 10.1109/ICMLA.2017.0-169

Yan, Y., Hashemi, M., Swersky, K., Yang, Y. & Koutra, D. (2022, November). Two Sides of the Same Coin: Heterophily and Oversmoothing in Graph Convolutional Neural Networks. In (pp. 1287–1292). doi: 10.1109/ICDM54844.2022.00169

Yang, G., Yang, Q. & Jin, H. (2021, September). A novel trust recommendation model for mobile social network based on user motivation. *Electronic Commerce Research*, *21*(3), 809–830. doi: 10.1007/s10660-019-09344-9

Yang, T., Wang, Y., Yue, Z., Yang, Y., Tong, Y. & Bai, J. (2022, June). Graph Pointer Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, *36*, 8832–8839. doi: 10.1609/aaai.v36i8.20864

Yao, L., Wang, X., Sheng, Q. Z., Benatallah, B. & Huang, C. (2021, March). Mashup Recommendation by Regularizing Matrix Factorization with API Co-Invocations. *IEEE Transactions on Services Computing*, *14*, 502–515. doi: 10.1109/TSC.2018.2803171

Yao, L., Wang, X., Sheng, Q. Z., Ruan, W. & Zhang, W. (2015, June). Service Recommendation for Mashup Composition with Implicit Correlation Regularization. In (pp. 217–224). doi: 10.1109/ICWS.2015.38

Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L. & Leskovec, J. (2018, July). Graph Convolutional Neural Networks for Web-Scale Recommender Systems. , 974–983. doi: 10.1145/3219819.3219890

Yoon, M., Jin, W. & Kang, U. (2018, April). Fast and Accurate Random Walk with Restart on Dynamic Graphs with Guarantees. In (pp. 409–418). Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee. doi: 10.1145/3178876.3186107

You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z. & Shen, Y. (2020). Graph Contrastive

Learning with Augmentations. In (Vol. 33, pp. 5812–5823). Curran Associates, Inc.

Yu, B., Yin, H. & Zhu, Z. (2018, July). Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In (pp. 3634–3640). (arXiv: 1709.04875 [cs]) doi: 10.24963/ijcai.2018/505

Yu, J., Xia, X., Chen, T., Cui, L., Hung, N. Q. V. & Yin, H. (2023). XSimGCL: Towards Extremely Simple Graph Contrastive Learning for Recommendation. , 1–14. doi: 10.1109/TKDE.2023.3288135

Yu, J., Yin, H., Li, J., Wang, Q., Hung, N. Q. V. & Zhang, X. (2021, June). Self-Supervised Multi-Channel Hypergraph Convolutional Network for Social Recommendation. In (pp. 413–424). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3442381.3449844

Yu, J., Yin, H., Xia, X., Chen, T., Cui, L. & Nguyen, Q. V. H. (2022, July). Are Graph Augmentations Necessary? Simple Graph Contrastive Learning for Recommendation. In (pp. 1294–1303). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3477495.3531937

Yu, J., Yin, H., Xia, X., Chen, T., Li, J. & Huang, Z. (2023). Self-Supervised Learning for Recommender Systems: A Survey. , 1–20. doi: 10.1109/TKDE.2023.3282907

Yu, P., Bao, B.-K., Tan, Z. & Lu, G. (2024, July). Improving Graph Collaborative Filtering with Directional Behavior Enhanced Contrastive Learning. *ACM Trans. Knowl. Discov. Data*, *18*, 183:1–183:20. doi: 10.1145/3663574

Yu, T., Liu, H., Zhang, L. & Liu, H. (2023, May). MSRDL: Deep learning framework for service recommendation in mashup creation. *Scientific Reports*, *13*(1), 7641. (Publisher: Nature Publishing Group) doi: 10.1038/s41598-023-32814-y

Yu, W. & Qin, Z. (2020). Graph Convolutional Network for Recommendation with Low-pass Collaborative Filters. In *International Conference on Machine Learning*.

Yu, W., Zhang, Z. & Qin, Z. (2022). Low-Pass Graph Convolutional Network for Recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Yu, Z. & Gao, H. (2022, June). Molecular Representation Learning via Heterogeneous Motif Graph Neural Networks. In (pp. 25581–25594). PMLR.

Yuan, F., Karatzoglou, A., Arapakis, I., Jose, J. M. & He, X. (2019, January). A Simple Convolutional Generative Network for Next Item Recommendation. In (pp. 582–590). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3289600.3290975

Yuan, H., Yang, J. & Huang, J. (2022, July). Improving hypergraph convolution network collaborative filtering with feature crossing and contrastive learning. *Applied Intelligence*, *52*, 10220–10233. doi: 10.1007/s10489-021-03144-1

Zhai, D., Yan, C., Zhong, W., Ding, S., Qi, L. & Zhou, X. (2025, October). Counteracting Popularity Bias in Multimedia Web API Recommendation. *IEEE Transactions on Computational Social Systems*, *12*, 3858–3868. doi: 10.1109/TCSS.2024.3517601

Zhang, C., Han, Q., Tan, Q., Wang, S., Zhao, X. & Chen, R. (2025, July). DimCL: Dimension-Aware Augmentation in Contrastive Learning for Recommendation.

In (pp. 1913–1923). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3690624.3709200

Zhang, C., Song, D., Huang, C., Swami, A. & Chawla, N. V. (2019). Heterogeneous Graph Neural Network. In (pp. 793–803). New York, NY, USA: Association for Computing Machinery. (event-place: Anchorage, AK, USA) doi: 10.1145/3292500.3330961

Zhang, D., Geng, Y., Gong, W., Qi, Z., Chen, Z., Tang, X., ... Tang, J. (2024, May). RecDCL: Dual Contrastive Learning for Recommendation. In (pp. 3655–3666). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3589334.3645533

Zhang, H. & McAuley, J. (2020, January). Stacked Mixed-Order Graph Convolutional Networks for Collaborative Filtering. In (pp. 73–81). Society for Industrial and Applied Mathematics. doi: 10.1137/1.9781611976236.9

Zhang, S., Hu, Z., Subramonian, A. & Sun, Y. (2024, August). Motif-Driven Contrastive Learning of Graph Representations. *IEEE Transactions on Knowledge and Data Engineering*, *36*, 4063–4075. doi: 10.1109/TKDE.2024.3364059

Zhang, S., Yao, L., Sun, A. & Tay, Y. (2019, February). Deep Learning Based Recommender System: A Survey and New Perspectives. *ACM Comput. Surv.*, *52*, 5:1–5:38. doi: 10.1145/3285029

Zhang, W., Zhang, Y., Xu, L., Zhou, J., Liu, Y., Gu, M., ... Yang, S. (2019). Modeling IoT Equipment With Graph Neural Networks. , *7*, 32754–32764. doi: 10.1109/ACCESS.2019.2902865

Zhang, Y. & Chen, X. (2020, March). Explainable Recommendation: A Survey and New Perspectives. *Found. Trends Inf. Retr.*, *14*, 1–101. doi: 10.1561/1500000066

Zhang, Y., Yu, J., Liu, Z., Wang, G., Nguyen, M., Sheng, Q. Z. & Wang, N. (2025, February). Improving graph collaborative filtering with network motifs. , 1–20. doi: 10.1007/s00521-025-11079-8

Zhang, Y., Zhu, J., Zhang, Y., Zhu, Y., Zhou, J. & Xie, Y. (2024, June). Social-aware graph contrastive learning for recommender systems. *Applied Soft Computing*, *158*, 111558. doi: 10.1016/j.asoc.2024.111558

Zhang, Z., Cui, P. & Zhu, W. (2022, January). Deep Learning on Graphs: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, *34*(01), 249–270. doi: 10.1109/TKDE.2020.2981333

Zhang, Z., Liu, Q., Wang, H., Lu, C. & Lee, C.-K. (2021, December). Motif-based Graph Self-Supervised Learning for Molecular Property Prediction. *Advances in Neural Information Processing Systems*, *34*, 15870–15882.

Zhao, C., Yang, E., Liang, Y., Zhao, J., Guo, G. & Wang, X. (2025, May). Symmetric Graph Contrastive Learning against Noisy Views for Recommendation. *ACM Trans. Inf. Syst.*, *43*(3), 80:1–80:28. doi: 10.1145/3722103

Zhao, H., Xu, X., Song, Y., Lee, D. L., Chen, Z. & Gao, H. (2021, May). Ranking Users in Social Networks with Motif-Based PageRank. *IEEE Transactions on Knowledge and Data Engineering*, *33*(05), 2179–2192. doi: 10.1109/TKDE.2019.2953264

Zhao, H., Zhou, Y., Song, Y. & Lee, D. L. (2019, November). Motif Enhanced

Recommendation over Heterogeneous Information Network. In (pp. 2189–2192). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3357384.3358134

Zhao, R., Chen, L., Sun, S., Peng, J. & Ju, S. (2024, December). Data Collaborative Contrastive Recommendation model with self-adaptive noise. , *256*, 124899. doi: 10.1016/j.eswa.2024.124899

Zheng, X., Liu, Y., Pan, S., Zhang, M., Jin, D. & Yu, P. S. (2022, February). *Graph Neural Networks for Graphs with Heterophily: A Survey.* (Issue: arXiv:2202.07082 arXiv: 2202.07082 [cs]) doi: 10.48550/arXiv.2202.07082

Zheng, X., Zhang, M., Chen, C., Zhang, Q., Zhou, C. & Pan, S. (2023, April). Auto-HeG: Automated Graph Neural Network on Heterophilic Graphs. In (pp. 611–620). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3543507.3583498

Zheng, Y., Xu, J. & Chen, L. (2024, March). *Learn from Heterophily: Heterophilous Information-enhanced Graph Neural Network.* (Issue: arXiv:2403.17351 arXiv: 2403.17351 [cs])

Zheng, Z. & Lyu, M. R. (2013, March). Personalized reliability prediction of web services. *ACM Trans. Softw. Eng. Methodol.*, *22*(2). doi: 10.1145/2430545.2430548

Zhou, G., Zhu, X., Song, C., Fan, Y., Zhu, H., Ma, X., … Gai, K. (2018, July). Deep Interest Network for Click-Through Rate Prediction. In (pp. 1059–1068). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3219819.3219823

Zhou, K., Wang, H., Zhao, W. X., Zhu, Y., Wang, S., Zhang, F., … Wen, J.-R. (2020, October). S3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. In (pp. 1893–1902). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3340531.3411954

Zhou, S., Guo, Z., Aggarwal, C., Zhang, X. & Wang, S. (2022, August). *Link Prediction on Heterophilic Graphs via Disentangled Representation Learning.* (Issue: arXiv:2208.01820 arXiv: 2208.01820 [cs]) doi: 10.48550/arXiv.2208.01820

Zhou, X., Sun, A., Liu, Y., Zhang, J. & Miao, C. (2023, June). SelfCF: A Simple Framework for Self-supervised Collaborative Filtering. *ACM Transactions on Recommender Systems*, *1*, 9:1–9:25. doi: 10.1145/3591469

Zhou, X., Zhuang, F., Xu, C., Zhao, P., Liu, Y., Xu, J., … Fang, J. (2019). Graph Contextualized Self-Attention Network for Session-based Recommendation. In (pp. 3940–3946). Macao.

Zhou, Z., Xie, Q., Wang, Y., Li, L., Liu, Y. & Tang, M. (2024, May). Debiased Contrastive Learning For Graph Collaborative Filtering. In (pp. 48–54). doi: 10.1109/CSCWD61410.2024.10580171

Zhou, Z., Zhou, S., Mao, B., Chen, J., Sun, Q., Feng, Y., … Wang, C. (2024, June). *Motif-driven Subgraph Structure Learning for Graph Classification.* (Issue: arXiv:2406.08897 arXiv: 2406.08897 [cs]) doi: 10.48550/arXiv.2406.08897

Zhu, H. & Koniusz, P. (2020, September). Simple Spectral Graph Convolution..

Zhu, J., Rossi, R. A., Rao, A., Mai, T., Lipka, N., Ahmed, N. K. & Koutra, D. (2021, May). Graph Neural Networks with Heterophily. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*, 11168–11176. doi: 10.1609/aaai.v35i12.17332

Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L. & Koutra, D. (2020). Beyond Homophily in Graph Neural Networks: Current Limitations and Effective Designs. In (Vol. 33, pp. 7793–7804). Curran Associates, Inc.

Zhuang, C. & Ma, Q. (2018, April). Dual Graph Convolutional Networks for Graph-Based Semi-Supervised Classification. In (pp. 499–508). Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee. doi: 10.1145/3178876.3186116

Zhuo, J., Lu, Y., Ning, H., Fu, K., Niu, B., He, D., ... Yang, L. (2024, November). Unified Graph Augmentations for Generalized Contrastive Learning on Graphs..