الجمهورية الشعبية الديمقراطية الجزائرية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement supérieur et de la Recherche scientifique

المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

Ecole Nationale Polytechnique
Département Génie Industriel

End of Study Project Dissertation

for **Obtaining State Engineer's Degree**
in **Industrial Engineering**
Option : **Data Science & Artificial Intelligence**

Interpretable Recommender Systems: A Hybrid Architecture with Logical and Collaborative Filtering Layers

Presented by:

MAZARI BOUFARES Nadhir

Defended on **26 Sept, 2024**, before a jury composed of :

Mr. Hakim FOURAR LAIDI	President
Mrs. Sofia AIT BOUAZZA	Examiner
Ms. Samia BELDJOUDI	Supervisor

ENP 2024

الجمهورية الشعبية الديمقراطية الجزائرية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement supérieur et de la Recherche scientifique

المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

Ecole Nationale Polytechnique
Département Génie Industriel

End of Study Project Dissertation

for **Obtaining State Engineer's Degree**
in **Industrial Engineering**
Option : **Data Science & Artificial Intelligence**

Interpretable Recommender Systems: A Hybrid Architecture with Logical and Collaborative Filtering Layers

Presented by:

MAZARI BOUFARES Nadhir

Defended on **26 Sept, 2024**, before a jury composed of :

Mr. Hakim FOURAR LAIDI	President
Mrs. Sofia AIT BOUAZZA	Examiner
Ms. Samia BELDJOUDI	Supervisor

ENP 2024

الجمهورية الشعبية الديمقراطية الجزائرية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

Ecole Nationale Polytechnique
Département Génie Industriel

Mémoire de Projet de Fin d'Études

en vue de l'obtention du **Diplôme d'Ingénieur d'État**
en **Génie Industriel**
Option : **Science des Données & Intelligence Artificielle**

Systèmes de Recommandation Interprétables : Une Architecture Hybride avec des Couches Logiques et de Filtrage Collaboratif

Présenté par :

MAZARI BOUFARES Nadhir

Soutenu le **26 septembre 2024**, devant le jury composé de :

M. Hakim FOURAR LAIDI	Président
Mme Sofia AIT BOUAZZA	Examinatrice
Mme Samia BELDJOUDI	Encadrante

ENP 2024

الملخص

تتطور أنظمة التوصية بسرعة مع زيادة التخصيص لتلبية القيود الجديدة وتحسين الأداء على المنصات الرقمية. ومع ذلك، يظل نقص الشفافية في عملية اتخاذ القرار، خاصة في الأساليب السوداء الصندوق، مشكلة كبيرة. يوفر دمج المنطق وأساليب الرموز حلاً واعدًا لتعزيز الفهم، لكن هذه الأساليب غالبًا ما تكون غير مستغلة بشكل كافٍ. تقترح هذه الأطروحة نموذجًا جديدًا لنظام التوصية يعزز من قابلية الفهم للمستخدمين النهائيين. يتكامل نموذجنا مع طبقة منطقية لإنشاء قواعد من سمات المستخدم والعنصر، إلى جانب شبكة التفاف رسومية لتصفية المعلومات المشتركة. من خلال دمج هذه المكونات، ينتج نموذجنا درجات توصية مع شفافية وقابلية تفسير محسنة.

الكلمات المفتاحية --- نظام التوصية، المنطق، قابلية التفسير.

Résumé

Les systèmes de recommandation (SR) évoluent rapidement avec une personnalisation croissante pour répondre aux nouvelles contraintes et améliorer les performances sur les plateformes numériques. Cependant, un problème majeur persiste : le manque de transparence dans leur processus de prise de décision, en particulier avec les approches en boîte noire. L'intégration du raisonnement logique et des méthodes symboliques offre une solution prometteuse pour améliorer l'interprétabilité, mais ces méthodes sont souvent sous-exploitées.

Cette thèse propose un nouveau modèle de SR qui améliore l'interprétabilité pour les utilisateurs finaux. Notre architecture intègre une couche logique pour générer des règles à partir des attributs des utilisateurs et des articles, ainsi qu'un réseau de convolution graphique pour le filtrage collaboratif. En combinant ces composants, notre modèle génère des scores de recommandation avec une meilleure transparence et interprétabilité.

Mots-clés— Système de recommandation, Raisonnement, Interprétabilité.

Abstract

Recommender systems (RSs) are rapidly evolving with increasing personalization to meet new constraints and improve performance on digital platforms. However, a significant issue remains: the lack of transparency in their decision-making, particularly with black-box approaches. Integrating logical reasoning and symbolic methods offers a promising solution for enhancing interpretability, but these methods are often underutilized. This thesis proposes a novel RS model that enhances interpretability for end users. Our architecture integrates a logical layer for generating rules from user and item attributes, alongside a graph convolutional network for collaborative filtering. By combining these components, our model generates recommendation scores with improved transparency and interpretability.

Keywords— Recommendation System, Reasoning, Interpretability

Acknowledgement

In the name of Allah, the Most Gracious, the Most Merciful. All praise is due to Allah, Lord of the worlds, for granting me the strength, knowledge, and opportunity to undertake this research study and to persevere and complete it satisfactorily. Without His blessings, this achievement would not have been possible.

I would like to express my profound gratitude to my research supervisors, Professors Mme. Amel Bouzeghoub and Mme. Samia Beldjoudi, for their exceptional guidance, unwavering support, and profound expertise throughout my research journey. Their insightful feedback pushed me to sharpen my thinking and brought my work to a higher level. Their patience, motivation, and immense knowledge were instrumental in helping me navigate the challenges of academic research and writing.

Lastly, I wish to express my sincere gratitude to all those who have contributed to the successful completion of this thesis. Whether through academic support, technical assistance, or words of encouragement, your contributions have not gone unnoticed. This accomplishment is as much a result of your collective support as it is of my individual effort.

Contents

List of Figures

List of Tables

List of Acronyms

AI: Artificial Intelligence
ML: Machine Learning
NeSy: Neuro-Symbolic
DL: Deep Learning
ReLU: Rectified Linear Unit
MLP: Multi-Layer Perceptron
MSE: Mean Squared Error
RSs: Recommendation Systems
TF-IDF: Term Frequency-Inverse Document Frequency
ROC: Receiver Operating Characteristic
CTR: Click Through Rate
CNN: Convolution Neural Network
GNNs: Graph Neural Networks
MPNNs: Message Passing Neural Networks
GCN: Graph Convolutional Network
GAT: Graph Attention Network
MCCF: Multi-Component Graph Convolutional Collaborative Filtering
NDCG: Normalized Discounted Cumulative Gain
GraphSAGE: Graph Sample and AggregatE
BPR: Bayesian Personalized Ranking
RRL: Rule Representation Learning
XAI: Explainable Artificial Intelligence
LIME: Local Interpretable Model-agnostic Explanations
NSXCF: The Neuro-Symbolic Collaborative Filtering Based Explanation
LightGCN: Light Graph Convolutional Network
NGCF: Neural Graph Collaborative Filtering
NSICF: Neuro-Symbolic Interpretable Collaborative Filtering
MLLP: Multilayer Logical Perceptrons

General Introduction

Context

As digital services are growing to meet the developing needs of the business and people, their performance requirements are growing as well, A good of amount of these services try to display content to users so the user can choose from, these type of services usually integrates recommendation engine into their service so it can display personalized content to the user, And good digital services is service who can provide their user of what they want, this is where recommendation comes to play, where a lot of digital services like music videos and commerce providers display a list of items that the user want to select, intuitively the digital service need to display what they user want to choose from all of the items the service provide, and these recommended items not only need to be relevant to the user but also need to be new to they user,diverse, and more.

Our Work focus on the recommendation system and their requirements, specifically explainablility, specially after their architecture are getting more complicated to provide accurate recommended items, explainability and interpretability also need to be considered this is when digital service also provide explanation of why certain items got recommended and others did not. this not only could help gaining the user trustworthiness but the service also can know more about their service and what they provide.

Problem Statement

This thesis aims to address the following research questions: How can we construct a recommendation model that effectively recommends items to users while providing clear explanations for the final output? And how can we evaluate the effectiveness of this explainability?

To construct a robust model, several challenges must be addressed:

- Integrating interpretability into the recommendation task, which centers around users and items, requires exploiting interpretable architectures specific to this task that are very rare in the research field.
- Ensuring that modifications for enhancing explainability do not deteriorate the model's recommendation accuracy.
- Developing explanation styles that satisfy end-user needs.

Thesis Outline

This engineering thesis is structured into two main parts:

- ***Part One:*** Literature Review of Recommendation Systems
  - **Chapter 1:** Overview of Recommender Systems
    A review of traditional approaches like content-based filtering and collaborative filtering, along with recent advancements in machine learning for recommendation systems.
  - **Chapter 2:** Graph-Based Approaches
    Exploration of graph-based methods as a growing trend in recommendation systems.
  - **Chapter 3:** Symbolic and Logic-Based Techniques
    Examination of how symbolic reasoning and logic can be integrated into recommendation systems.
- ***Part Two:*** Proposed Approach and Experimental Validation
  - **Chapter 4:** Proposed Architecture
    Introduction of the proposed recommendation system architecture, detailing the various layers and activation functions used.
  - **Chapter 5:** Experimentation
    Description of the experimental setup, including datasets, metrics, and baseline models for comparison and the analysis of the results, including performance comparison with other models, ablation studies, and interpretation of findings.

Literature Review

Chapter 1

Recommendation Systems

1.1 Introduction

Recommendation Systems are algorithms that leverage user preferences and historical data to generate personalized recommendations for a wide range of items and services. In this chapter, we will discuss how rs are studied under different conceptual frameworks. We will list these concept types, explore their interrelationships, and examine the approaches used to develop recommendation systems.

In the first section, we will cover the primary concept of recommendation engines: the application of RS, including the data utilized in their development and their corresponding applications. The second section will describe the different stages of recommendation, which can also be referred to as approaches. In the third section, we will discuss various recommendation system scenarios. Finally, we will address the evaluation concepts and objectives of RS.

As a **definition**, Recommendation Systems or engines are filtering tools that suggest specific items to users. Various types of recommendation systems have been developed based on the data used, the application, the scenario, the approach, and the objective [1].

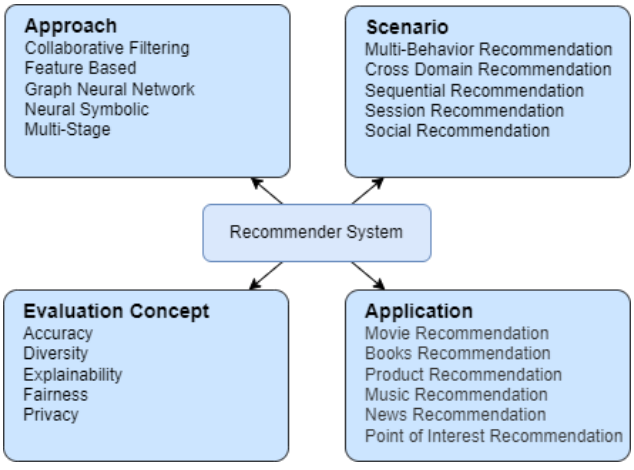

Figure 1.1: Recommendation systems Concepts

As we can see in the figure 1.1, there are four concept types that Recommendation Systems

can be studied over:

- **Applications Concepts:** These are determined by the content of items and users. Applications can include movies, videos, music, books, and social posts.
- **Evaluation Concepts:** This refers to what we aim to achieve from the recommender systems. It can be accuracy as well as other metrics like diversity and explainability, or a combination of these.
- **Scenario Concepts:** This reflects the environment that the recommender system operates in. Examples include cross-domain recommendation, which uses multiple applications (like music, movies, and books) to provide recommendations, or multi-behavior scenarios, where the system predicts the user's next choice based on clicks and navigation behavior.
- **Approach:** This concerns to the methodology used to build the recommender systems. The goal is to satisfy as many criteria as possible from evaluation, application, and scenario perspectives.

In the following sections, we will see how each of these four concepts relates to the others, and we will study the relationships between different kinds of concepts.

1.2 Application

Recommendation applications can be studied under various scenarios, approaches, and evaluation concepts.

- How the scenarios affect the application and vice-versa : Some applications are domain-specific, such as music recommendation, while others span multiple domains, as seen in e-commerce websites.
- Viewing applications through the lens of evaluation concepts reveals nuances. For instance, medical recommendation systems demand a careful attention to accuracy and explainability, whereas social recommendation systems prioritize fairness.
- Regarding approaches, in contexts like e-commerce, fast and attribute-based approaches are often more pertinent than alternative methods like graph neural network.

When we say the application of the recommendation, we are referring directly to the data content that we use for building the RS, some platform usage data can be used directly to build recommendation engines, other external data (like knowledge graph data) can be incorporate to add more attributes that gives more meanings.

One way to represent the recommendation data, specifically the interaction data, is by user*item matrix relationship that can be drown between these two features can be built as represented below.

	Item 1	Item 2	Item 3
User 1	4	0	2
User 2	3	4	1
User 3	0	5	0

Table 1.1: Visual Representation of Item User Matrix

The values of the matrix can represent the rating between user and item that can help filter.

this relationship between items and users be calculated explicitly or implicitly.

- **Explicit Feedback**: Scalar Ratings, Textual Reviews, Favorites,Binary Rating, Tags.
- **Implicit Feedback**: Click Through Rate (CTR), Search History, Purchase History.

Data Sources

In research, data sources can originate from various entities. Some are provided by companies as subsets of their platforms, such as Netflix. Others are collected by research labs and groups, like GroupLens. Certain datasets are domain-specific (e.g., books), while others span multiple domains. Some datasets are developed for specific scenarios, such as behavior recommendation or sequential recommendation. The figure below highlights the most commonly used open-source datasets for developing and testing RS under different conceptual frameworks.

In this table, we present various well-known datasets used in recommendation systems. These datasets can be characterized by application, the number of users and items, as well as the number of interactions between them. If users interact frequently with items, the dataset is considered dense. Additionally, datasets may provide attributes related to users, items, or the relationships between them.

Dataset Name	Description	Users	Items
Yelp2018 [2]	From the 2018 edition of the Yelp challenge. Local businesses like restaurants and bars are viewed as items.	1,987,897	150,346
MovieLens	Structured with multiple size variations provided by the research group Grouplens.	–	–
Netflix Prize	From an open competition for the best collaborative filtering algorithm to predict user ratings for films, without any other information about the users or films.	–	94,000
IMDB Data	Sub dataset of the platform containing users and ratings data.	–	–
Dianping Restaurant Reviews	With two versions: one suited for sequential recommendation and another for social recommendation.	616,331	10,979
LibraryThing	Collected from different book websites by the LibraryThing platform.	73,882	337,561
Epinions	From an online social network of a general consumer review site used for social recommendation.	116,260	41,269
Taobao User Behaviour	User Behavior dataset from Taobao, for recommendation problems with implicit feedback. Provided by Alimama.	987,994	4,162,024
Amazon	Large-scale Amazon Reviews dataset collected in 2023 by the University of California, San Diego.	54,510,000	48,190,000

Table 1.2: Dataset descriptions

In the table below 1.3, we list some external data sources that can be useful for recommendation systems. These external sources provide additional data about the content of the items being recommended. Incorporating this data can enhance the representation of items, leading to more personalized RS.

Dataset Name	**Description**
DBpedia	Crowd-sourced community information created in various Wikimedia projects such as Wikipedia. It is a Knowledge Graph consisting of over 5 billion facts, which are represented using RDF format, and linked to external Knowledge Graphs such as Freebase and Wikidata.
LinkedMDB	A knowledge graph for movies, actors, directors, and other film-related data, providing structured information from the Linked Open Data cloud.
Wikidata	A collaboratively edited knowledge graph hosted by the Wikimedia Foundation, providing a central storage for structured data of its Wikimedia sister projects.
YAGO	A large semantic knowledge base derived from Wikipedia, WordNet, and GeoNames, integrating information about millions of entities and their relationships.
Freebase	A community-curated database of structured data, acquired by Google, which integrates data from various sources and is used to enhance Google's Knowledge Graph.

Table 1.3: External Data Descriptions

1.3 Approach

Approach refer to the computational algorithm utilized in Recommendation system or tasks to serve specific scenarios or multiple scenarios with good performance in evaluation concepts. The primitive approaches, like content-based filtering and collaborative filtering, have not performed well in computational evaluation and suffer from issues such as the cold start problem [3].

Together with the evaluation concepts, some challenges like the cold start problem and sparsity are addressed when developing new approaches. Sparsity occurs when many data points are empty, significantly impacting our recommender system, while the cold start problem arises when the approach fails to consider new users and/or new items.

Below, we present different approach concepts which can overlap in hybridization techniques or in multi-staging.

1.3.1 Non Personalized

A non-personalized recommendation system can be defined as an information retrieval method that suggests items to users without tailoring the suggestions to their individual characteristics or past behaviors, but only looking the popularity of the items. so they suggest the most suggested items to the all of the users without taking the interaction data between specific user and the items nor the use attributes. it can help reduce the effects of the cold start problem for the approaches who can't handle the challenge like collaborative filtering.

1.3.2 Content Based

Content-based recommender systems are designed to suggest items to users based on the attributes of the items and the user's interaction history. This approach focuses on analyzing the features of items that a user has previously interacted with to predict new items they might like. The system creates a user profile based on these interactions, which is then used to compare and recommend similar items, without needing data from other users [4].

Key Features

- **User Profile Creation** : The user profile is generated by collecting data from user interactions such as purchases, ratings, and search history. This profile is crucial for understanding user preferences and tailoring recommendations accordingly.
- **Item Feature Analysis** : Content-based systems analyze the features of items, which can be keywords, genres, or other descriptive attributes. These features are used to compare items and identify those that are similar to what the user has previously liked.
- **Similarity Metrics** : A key component of content-based filtering is the use of similarity metrics to compare user profiles with item features. Common metrics include

cosine similarity, dot product, and others, which help in scoring and ranking items for recommendation.

Types of Content

- **Content** : This involves using keywords like movie genres to describe items. The system matches these keywords with user preferences to suggest similar items.
- **Textual Content** : Textual analysis involves examining the text associated with items, such as the plot of a movie. Techniques like Term Frequency Inverse Document Frequency are used to process and analyze textual data.
- **Semantic Content** : This involves understanding the deeper semantic relationships between items, often using ontologies or semantic networks to enhance the recommendation process.

1.3.3 Collaborative Filtering

the concept is that a user's rating for a new item is likely to resemble that of another user if they have similarly rated other items. Likewise, if multiple users have given similar ratings to two items, it's probable that a user will rate those items similarly as well [5]. There are two primary approaches to collaborative filtering:

- **The User-Based Approach**: This method involves comparing users to each other and identifying those with similar preferences. A user's rating is then predicted based on the ratings of users within their "neighborhood."
- **The Item-Based Approach**: This approach entails grouping together items that are liked by the same users and predicting user ratings based on items that are similar to those they have previously rated.

Matrix Factorization

A technique used to solve the problem of sparsity in user-item matrix when performing the collaborative filtering by decomposing the matrix into more relevant features vectors called latent vectors, for example singular value decomposition is an algorithm that can perform matrix factorization [6].
Note : Matrix Factorization can be also used in item-feature matrix decomposition to incorporate later this matrix alongside the user-item matrix.

1.3.4 Hybrid Approaches

With different approaches present different problems and lacks, a different technique of hybridization were proposed, though the hybridization take two phases.

In the phase one we perform item to user filtering independent.

In the phase two, Combine these sets of recommendations through hybridization methods such as weighting, mixing, cascading, switching.

Hybridization Methods

The **switching** hybridization technique involves the system switching between different recommendation techniques depending on the current situation or user context. This allows the system to leverage the strengths of different techniques to overcome issues like cold-start and data sparsity [7].

Mixed hybrid recommender systems present the results from multiple recommendation techniques simultaneously to the user. This can help avoid issues like cold-start and data sparsity, but may still suffer from data over-specialization since multiple recommenders are providing their individual results.
In a **cascading** hybrid, one recommendation technique is first used to produce a coarse candidate list of items, which is then refined by applying other recommendation techniques.
In a **weighted** hybrid recommender system, the score or weight of a recommended item is calculated as a combination of the results from all the available recommendation techniques implemented in the system.

1.4 Scenario

In recommendation systems, various scenarios arise depending on the context and the specific of the platforms and how they deliver their content to the user. Each scenario leverages different types of data and user interactions to tailor recommendations effectively. This section outlines the primary scenarios in which recommendation systems operate [1].

1.4.1 Social Recommendation

Social recommendation systems utilize social connections and interactions to suggest items of interest. These systems capitalize on the preferences and activities of a user's friends or connections on social media platforms to provide personalized recommendations. For example, a social recommendation system might suggest movies or books that friends have liked or shared, leveraging the trust and influence inherent in social networks.

1.4.2 Multi-behavior Recommendation

Multi-behavior recommendation systems consider a variety of user behaviors, such as purchases, ratings, and likes, to generate personalized recommendations. By integrating multiple behavioral signals, these systems can create a more comprehensive profile of user preferences. For instance, an e-commerce site might suggest products based not only on past purchases but also on items the user has viewed, rated, or added to their wishlist.

1.4.3 Cross-domain Recommendation

Cross-domain recommendation systems extend their recommendations across different categories or domains. These systems transfer knowledge from one domain to another to provide holistic recommendations. For example, a system might recommend movies based on a user's music preferences or suggest books based on the user's favorite television shows. This approach leverages the interconnectedness of different domains to enhance the recommendation quality.

1.4.4 Sequential Recommendation

Sequential Recommendation System analyze the order and timing of user interactions to make informed recommendations. By considering the sequence of user actions, these systems can predict the next item of interest more accurately. For instance, in a music streaming service, a sequential RS might suggest the next song in a playlist based on the user's listening history and patterns, ensuring a smooth and enjoyable listening experience.

1.4.5 Session-based Recommendation

Session-based recommendation systems focus on providing real-time recommendations within a single user session. They analyze user actions, such as clicks and searches, during the current session to suggest relevant items on-the-fly. For example, as a user navigates a news website, a session-based recommendation system might recommend related articles based on the user's current reading interests, enhancing the browsing experience by delivering timely and contextually relevant suggestions.

1.5 Evaluation

As the same for the other approach's,in the literature of recommendation systems, the evaluation process is multifaceted, shaped by the difference scenarios, approaches and applications and challenges [1]. Various evaluation aspects have been defined in the research, encompassing utility, diversity, and the intricate interplay between these dimensions. However, our focus in this section see each concept independently then we see how the relationship between the concepts are studied. some evaluations are not defined heavily like trustworthiness and risk.

It's noteworthy that older surveys often overlooked the aspect of explainability, primarily due to the pervasive dominance of classical recommendation approaches in application scenarios [8]. Nonetheless, the evolving research has witnessed a notable shift, with recent surveys dedicated exclusively to study explainability in recommendation systems [9].

Some concept like diversity and novelty are studied on different *levels* of recommendations (Life Level, System Level, Recommendation Level), the levels were organized according to human activity as whole.

At the final section we discuss different approaches to evaluate the recommendation system beside the most used approach of offline evaluation

1.5.1 Accuracy

Error metrics are widely used for predictive accuracy.

Mean Absolute Error evaluates the difference between the ratings. predicted by the recommended and given by the users [5]. Equation 1.1 show the MAE metric.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{1.1}$$

Root Mean Squared Error is another error metric, Root Mean Squared Error calculates a larger difference for large errors in the rating prediction [10]. as it is shown in 1.2.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{1.2}$$

Both MAE and RMSE are calculated on the prediction list, therefore the metrics are divided by R_u. In addition, there are other error metrics, such as Average RMSE, Average MAE and Mean Squared Error.

Precision and Recall

According to [10], **precision** of a recommendation consists on the number of consumed (or rated) items C_u in the recommendation list, as stated in the 1.3. Precision measures the rate of items in the recommendation list that the user likes and therefore consumed.

$$\text{util}(R_u) = \text{precision} = \frac{|C_u \cap R_u|}{|R_u|} \tag{1.3}$$

According to [10], **Recall** R_u, on the other hand, is calculated by the number of consumed items C_u in the recommendation list out of the total number of items the user consumed. 1.4 show recall calculation. Authors have called precision and recall as precision@N and recall@N, where N stands for the size of the recommendation list.

$$\text{util}(R_u) = \text{recall} = \frac{|C_u \cap R_u|}{|C_u|} \tag{1.4}$$

Receiver Operating Characteristic

The Receiver Operating Characteristic calculates The rate of items that the user likes in the recommendation list. Differently from error, precision and recall metrics, the calculation of ROC curves accentuate items that were suggested but the user disliked. Evaluation of algorithms in different scenarios could use the Area under the ROC curve (AUC) [10].

Ranking Score

Recommenders usually predicts ranked lists, however, users difficultly browse through all of the items. Therefore, ranking metrics could be interesting in measuring the utility and rank information altogether.
One example **R-Score** is the metric which considers a deduction in the value of recommendations according to the rank position as the equation 1.5. Top ranked items are more valued rather then items in the tail of the metric [10].

$$\text{util}(R_u) = \text{rank}(R_u) = \sum_{j=1}^{|R_u|} \frac{\max(r(i_j) - d, 0)}{2^{\frac{j-1}{a-1}}} \tag{1.5}$$

Other ranking metrics are Kendall and Spearman rank correlation and Normalized Distance-based Performance Measure [5].

Click through Rate

Click through Rate CTR calculates the ratio of clicked/interacted recommended items out of the number of items recommended. It has been used since the early stages of the web in web/mobile advertisement and online marketing campaigns. CTR is also a major metric applied in the industry of recommender systems, as it helps to study how many items recommended to the users that they effectively consume.

1.5.2 Diversity

Diversity is concerned with divers items in the recommendation list. denoted as $div(R_u)$. In the literature, diversity was defined similarly not like other concepts as Novelty, according to [10], diversity has an opposite effect of similarity.

As a result of this definition, the proposed metrics tend to calculate diversity as a dissimilarity between the items in the recommendation list with intra-list similarity metric as 1.6 shows. meaning the lower values has more divers.

$$\text{div}(R_u) = \sum_{i \in R_u} \sum_{j \in R_u, i \neq j} d(i, j) \tag{1.6}$$

1.5.3 Unexpectedness

Evolved form serendipity as component [11] than to a concept, with increasing mentioning in literature however are still to be mentioned.

The metrics can be grouped into two group based on Primitive recommendation that evaluate the recommendation list provided directly but the recommended system, however the second group the non primitive based metrics compare the recommendation list with the user history.

1.5.4 Novelty

Defined at different levels, There are some authors that define novelty in the life level. as unknown items as never consumed or known in the users' lifetime.the system level define a novel item for a user is one that the user has none or little knowledge about. and Level 3 involves novelty in the recommendation list level, that is, items not repeatedly recommended. In this sense, novelty is defined as not repeated items in the recommendation list, not involving users' information [10].

1.5.5 Serendipity

The term serendipity means a lucky finding or a satisfying surprise. serendipity represent surprising recommendations. Metrics have been proposed to measure serendipity in recommendation lists and most of them have some relation to the concepts that serendipity is involved to: level 2 of novelty, unexpectedness and utility.

1.5.6 Coverage

Coverage is another concept that has been analyzed by previous researches in recommender systems. Although there have been few studies proposing metrics for coverage, it still worthy to mention due to its potential relation with the other explored concepts in this thesis. In addition, other concepts such as trust, risk, robustness are far less studied than coverage. The notation of coverage used in this thesis is *cov*. Coverage evaluates the whole RS, not a recommendation list. Moreover, three kinds of coverage are mentioned in the literature: item space coverage and user space coverage, as presented by [10].

1.5.7 Comprehensibility

Comprehensibility can be measured for different approaches and becoming more and more important with the incorporating new approaches like the deep neural network in the recommendation engines that put the interest of the user.

For the metrics are categorized into two approaches, One is to evaluate the percentage of recommendations that can be explained by the explanation model, regardless of the explanation quality; and the second approach is to evaluate the explanation quality directly.

Non Quality Metrics

Explain-ability precision (EP) is an example of non quality metric that is defined as the proportion of explainable items in the top-n recommendation list, relative to the total number of recommended (top-n) items for each user. **Explainability recall** (ER), on the other hand, is the proportion of explainable items in the top-n recommendation list, relative to the total number of explainable items for a given user [9], the disadvantage of this is that the model itself doesn't decide if an item is explainable or not but third party (a user or outside system).

Mean explainability precision (MEP) and mean explainability recall (MER) are EP and ER averaged across all testing users, respectively.
The idea was generalized to create **Fidelity** Metric [12] as the equation 1.7, although the metric suffered from the same issue as the MEP & MER.

$$Fidelity(R_u) = \frac{|\text{explainable items} \cap \text{recommended items}|}{|\text{recommended items}|} \tag{1.7}$$

Quality Metrics

For the second approach, evaluating the quality of the explanations usually depends on the type of recommendation system model, approaches and it's use case and tasks.
An Example commonly used explanation type is a piece of explanation sentence. In this case, offline evaluation can be conducted with text-based measures. For example, in many online review websites (such as e-commerce), we can consider a user's true review for an item as the ground-truth explanation for the user to purchase the item.

1.5.8 Evaluation Approaches

When we study recommendation concepts, we didn't focus on the approach used. Depending on the approach, different concepts and their metrics could be applied or not. the approaches that we will present can overlap to the study of specific experimentation [1].

- **Offline and Online evaluations** : Evaluations can be evaluated both online and offline. Usually, offline evaluation is easier to implement, since online evaluation and user studies would depend on the availability of data and users in real-world systems, which are not always accessible to researchers. As a result, online evaluation is encouraged but not always required for explainable recommendation research.
- **User Studies Evaluation** : Small scale. A few dozens or hundreds of users are presented recommendations created by different recommendation approaches, and then the users judge which recommendations are best.
- **A/B tests** : Recommendations are shown to typically thousands of users of a real product, and the recommendation system randomly picks at least two different recommendation approaches to generate recommendations. The effectiveness is measured with implicit measures of effectiveness such as conversion rate or CTR.
- **Primitive RS evaluation** : here we use the model output to evaluate the recommended list unlike the non primitive evaluation approach where we use the user history.

1.6 Conclusion

In conclusion, Recommender Systems play a pivotal role in filtering vast amounts of information to offer personalized suggestions, catering to diverse applications and domains.

Throughout this chapter, we explored key conceptual frameworks that guide the development of recommendation systems, including application, scenario, approach, and evaluation perspectives. We also examined various techniques such as content-based filtering, collaborative filtering, and hybrid approaches, each addressing specific challenges like the cold-start problem and data sparsity. By utilizing a combination of these methods, recommendation systems are continuously evolving to provide more accurate, relevant, and context-aware recommendations, contributing significantly to user satisfaction and engagement across a wide range of industries.

Chapter 2

Graph Neural Network Recommendation System

2.1 Introduction

Inspired by advancements in Convolutional Neural Networks (CNNs), Graph Neural Networks (GNNs) have been proposed as promising solutions for many problems involving data with a graph structure as we can find in recommendation systems. Graph Neural Networks are defined as a class of artificial neural networks designed for processing graph-structured data [10].

The first section will explore the application of GNNs in various scenarios. The second section will provide a generic pipeline of how the architecture of GNNs is developed, detailing the expected input, output, and processing steps. In the third section, we will discuss how comprehensive GNNs are constructed. The fourth section will examine the specific case of GNNs being applied to recommendation systems, explaining the developed approaches of RS using GNNs and how to evaluate them.

2.2 Graph Neural Network Applications

In a general sense, Graph Neural Network models can perform multiple tasks. Any application that satisfies both the required task and the necessary graph construction input data can be executed. In the literature, there is significant concentration on applications in the biological, medical, and recommendation fields, considering the interlink and data structure of these applications [13].
Below are some of the different application of GNNs 2.1 [14].

Area	Application
Biology	Protein Interface Prediction
	Disease Classification
Chemistry	Chemical Reaction Prediction
Recommendation System	User-item Interaction Prediction
Text	Text Classification
	Sequence Labeling
	Relation Extraction
Image	Image Classification
	Region Classification
	Visual Question Answering

Table 2.1: GNN Applications

2.3 Graph Neural Network Pipeline

In this section, we present the general design pipeline of a GNN model for a specific task on a specific graph type. Generally, the pipeline contains three steps: (1) find graph structure and specify graph type and scale, (2) design loss function and (3) build model using computational modules. We give general design principles and some background knowledge in this section.

2.3.1 Graph Constructing

At first, we have to find out the graph structure in the application. There are usually two scenarios: structural scenarios and non-structural scenarios. In structural scenarios, the graph structure is explicit in the applications, such as applications on molecules, physical systems, knowledge graphs and so on. In non-structural scenarios, graphs are implicit so that we have to first build the graph from the task, such as building a fully-connected "word" graph for text or building a scene graph for an image. After we get the graph, the later design process attempts to find an optimal GNNs model on this specific graph.

Constructing different types of graphs necessitates either pre-existing graph data like social data or abstracting the concept of graph, nodes and edges from non-structured data like textual data using various techniques like TFIDF.

And To have the computational mathematical representation of the graph, adjacency matrix, adjacency list, edge list can be used depending on the graph model requirement that it will be used.

Graph Type

these are different type that the graph can be described.

- **Directed/Undirected** Graphs. Edges in directed graphs are all directed from one node to another, which provide more information than undirected graphs. Each edge

in undirected graphs can also be regarded as two directed edges.

- **Homogeneous/Heterogeneous** Graphs. Nodes and edges in homogeneous graphs have same types, while nodes and edges have different types in heterogeneous graphs. Types for nodes and edgesplay important roles in heterogeneous graphs and should be further considered.
- **Static/Dynamic** Graphs. When input features or the topology of the graph vary with time, the graph is regarded as a dynamic graph. The time information should be carefully considered in dynamic graphs.
- **Hyper-graph**: A generalization of a graph where edges can connect more than two nodes.

Note : these categories are orthogonal, which means these types can be combined, e.g. one can deal with a dynamic directed heterogeneous graph. There are also several other graph types designed for different tasks such as hypergraphs and signed graphs. We will not enumerate all types here but the most important idea is to consider the additional information provided by these graphs. Once we specify the graph type, the additional information provided by these graph types should be further considered in the design process.

2.3.2 Network Loss Function

These optimization algorithms are developed based on the specific GNN application task To optimize the graph neural network models, the traditional loss functions always turn to graph learning losses. For example, the log loss in the optimization can be regarded as the point-wise link prediction loss. Similarly, **Bayesian Personalized Ranking (BPR)**loss [15] is usually adopted in the link prediction task on graphs.

In addition, sometimes, GNN-based recommendation may involve multiple tasks, such as the link prediction tasks on different types of edges. Then, in such a case, how to balance each task and make them enhance each other is challenging.

Bayesian Personalized Ranking

The Bayesian Personalized Ranking (BPR) loss function is commonly used in recommendation systems, particularly for collaborative filtering tasks. It is designed to optimize models that aim to rank items for each user based on their preferences. Here's the mathematical description of the BPR loss:

Let's assume we have a set of observed user-item interactions represented as triplets of the form (u, i, j), where u is a user, i is a positively rated item, and j is a negatively rated item. The goal is to learn a model that assigns higher scores to positively rated items compared to negatively rated items.

The BPR loss function is defined as follows 2.1:

$$\mathcal{L}_{\text{BPR}} = -\sum_{(u,i,j)\in\mathcal{D}} \log \sigma(\hat{r}_{uij}) \tag{2.1}$$

where:

- $\mathcal{D}$ is the set of observed user-item interactions.
- $\hat{r}_{uij}$ is the predicted preference score of user u for item i over item j.
- $\sigma(x)$ is the sigmoid function, defined as $\sigma(x) = \frac{1}{1+e^{-x}}$.

The preference score $\hat{r}_{uij}$ can be computed using a model, such as a matrix factorization model, neural network-based model, or any other recommendation model. The model aims to learn representations of users and items such that the preference score for a positively rated item i for user u is higher than that for a negatively rated item j for the same user u.

The BPR loss function encourages the model to assign higher preference scores to positively rated items compared to negatively rated items. Minimizing this loss function during training leads to better rankings of items for each user, improving the overall performance of the recommendation system.

Sampling

In recommendation systems, sampling is essential for efficiently evaluating the loss function, especially with large datasets. Instead of computing the loss across all user-item pairs, sampling selects a representative subset, reducing computational demands while maintaining accuracy. Negative sampling, a key technique, focuses on selecting negative interactions—pairs unlikely to occur—to help models differentiate between positive and negative cases. This method is widely used in algorithms like matrix factorization and neural networks, but its effectiveness depends on carefully choosing negative samples, as poor selection can impair model performance and generalization.

2.3.3 Computational Module

The generic implementation of GNNs can be represented as follows, in first we have message pooling or the graph model, that can be categorized on spectral and spatial format. different model were proposed like Graph Convolutional Networks, Graph Attention Networks.

Message Passing

Message Passing: In the message passing phase, information is exchanged between neighboring nodes in the graph. Each node aggregates information from its neighbors and updates its own representation based on this aggregated information. This step allows nodes to incorporate information from their local neighborhood.

Local Pooling: After message passing, in the local pooling phase, the graph may be downsampled or coarsened to reduce its size while preserving important structural information. This downsampling helps in reducing the computational complexity of the network by focusing on the most relevant parts of the graph.

Global Pooling: In the global pooling phase, the representation of the entire graph is computed. This can involve aggregating information from all nodes in the graph to produce a single vector representation that captures the overall characteristics of the graph. Global pooling enables the network to make predictions or classifications at the graph level.

It's worth mentioning that even though in global and local pooling aggregation are also made like in message passing but they serve different purpose than the message passing aggregation function. some task require carefully consideration into the pooling of the graph while other require more attention to the message passing specially when searching for representative architecture like in recommendation system.

the key data processing element of the GNNs architecture, which are defined as layers mapping a graph into an updated representation of the same graph. Formally, they can be expressed as Multi Logical Neural Network (MPNN).

Let $G = (V, E)$ be a graph, where V is the node set and E is the edge set. Let N_u be the neighbourhood of some node $u \in V$. Additionally, let $\mathbf{x}_u$ be the features of node $u \in V$, and $\mathbf{e}_{uv}$ be the features of edge $(u, v) \in E$. An MPNN layer can be expressed as follows 2.2:

$$\mathbf{h}_u = \phi\left(\mathbf{x}_u, \bigoplus_{v \in N_u} \psi(\mathbf{x}_u, \mathbf{x}_v, \mathbf{e}_{uv})\right) \tag{2.2}$$

Where ϕ and ψ are differentiable functions (e.g., artificial neural networks), and $\bigoplus$ is an aggregation operator that can accept an arbitrary number of inputs (e.g., element-wise sum, mean, or max). In particular, ϕ and ψ are referred to as update and message functions, respectively. Intuitively, in an MPNN computational block, graph nodes update their representations by aggregating the messages received from their neighbours.

The outputs of one or more MPNN layers are node representations $\mathbf{h}_u$ for each node $u \in V$ in the graph. Node representations can be employed for any downstream task, such as node/graph classification or edge prediction.

Initial Embedding Nodes that will get processed by the GNNs before feeding them into the architecture, initialization need to be considered this is where different approaches can be listed :

- In some cases, Node embedding are initialized with the **Identity Matrix**, which represents the initial featureless graph.
- **Node Attributes** If nodes have associated attribute information (e.g., node features, categorical attributes), node embedding can be initialized using these attributes.

- **Zero Initialization** : Node embedding are initialized with zero vectors. While this approach is straightforward, it may not be suitable for tasks where non-zero initialization are preferred.
- **Xavier initialization** [16], also known as Glorot initialization, is a technique used to initialize the weights of neural network layers in a way that helps to keep the gradients from exploding or vanishing during training.
- **Distribution initialization**, Using Normalization distribution to ensure fairness is also an often used method to initialize the embedding.

2.3.4 Examples

Graph Convolution Network Graph Convolution Network Architecture they get developed for specific applications the most promising being GCN which first introduced in 2018 are considered to be e generalization of CNN over graph data.

Most of GCN developed are considered to be spatial neural network because operates directly on the graph unlike the spectral where they leverage+e the graph Laplacian matrix or its eigenvectors to transform the graph data into a graph spectral representation.

A Graph Neural Network layer defines a first-order approximation of a localized spectral filter on graphs. GCN can be understood as a generalization of CNN to graph-structured data.

The formal expression of a GCN layer reads as follows 2.3:

$$\mathbf{H} = \sigma\left(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}\boldsymbol{\Theta}\right) \tag{2.3}$$

where $\mathbf{H}$ is the matrix of node representations $\mathbf{h}_u$, $\mathbf{X}$ is the matrix of node features $\mathbf{x}_u$, $\sigma(\cdot)$ is an activation function (e.g., ReLu), $\tilde{\mathbf{A}}$ is the graph adjacency matrix with the addition of self-loops, $\tilde{\mathbf{D}}$ is the graph degree matrix with the addition of self-loops, and $\boldsymbol{\Theta}$ is a matrix of trainable parameters.

In particular, let $\mathbf{A}$ be the graph adjacency matrix: then, one can define $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ and $\tilde{\mathbf{D}}_{ii} = \sum_{j \in V} \tilde{A}_{ij}$, where $\mathbf{I}$ denotes the identity matrix. This normalization ensures that the eigenvalues of $\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ are bounded in the range $[0, 1]$, avoiding numerical instabilities and exploding/vanishing gradients.

A limitation of GNC is that they do not allow multidimensional edge features $\mathbf{e}_{uv}$. It is however possible to associate scalar weights w_{uv} to each edge by imposing $A_{uv} = w_{uv}$, i.e., by setting each nonzero entry in the adjacency matrix equal to the weight of the corresponding edge.

Graph Attention Network Graph Attention Network is a combination of a graph neural network and an attention layer. The implementation of attention layer in graphical neural networks helps provide attention or focus to the important information from the data instead

of focusing on the whole data.

A multi-head GAT layer can be expressed as follows 2.4:

$$\mathbf{h}_u = \Big\|_{k=1}^{K} \sigma\left(\sum_{v \in N_u} \alpha_{uv} \mathbf{W}^k \mathbf{x}_v\right) \tag{2.4}$$

where K is the number of attention heads, $\|$ denotes vector concatenation, $\sigma(\cdot)$ is an activation function (e.g., ReLU), α_{ij} are attention coefficients, and $\mathbf{W}^k$ is a matrix of trainable parameters for the k-th attention head.

For the final Graph Attention Network layer, the outputs from each attention head are averaged before the application of the activation function. Formally, the final Graph Attention Network layer can be written as 2.5:

$$\mathbf{h}_u = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{v \in N_u} \alpha_{uv} \mathbf{W}^k \mathbf{x}_v\right) \tag{2.5}$$

Attention in Machine Learning is a technique that mimics cognitive attention. In the context of learning on graphs, the attention coefficient α_{uv} measures how important is node $u \in V$ to node $v \in V$.

Normalized attention coefficients are computed as follows 2.6:

$$\alpha_{uv} = \frac{\exp(\text{LeakyReLU}\left(\mathbf{a}^T[\mathbf{W}\mathbf{h}_u \| \mathbf{W}\mathbf{h}_v \| \mathbf{e}_{uv}]\right))}{\sum_{z \in N_u} \exp(\text{LeakyReLU}\left(\mathbf{a}^T[\mathbf{W}\mathbf{h}_u \| \mathbf{W}\mathbf{h}_z \| \mathbf{e}_{uz}]\right))} \tag{2.6}$$

Where $\mathbf{a}$ is a vector of learnable weights, $\cdot^T$ indicates transposition, and LeakyReLU is a modified ReLU activation function. Attention coefficients are normalized to make them easily comparable across different nodes.

A GCN can be seen as a special case of a GAT where attention coefficients are not learnable, but fixed and equal to the edge weights w_{uv}.

2.4 GNN Comprehensibility

As the compressible artificial intelligence field and interests are growing, ensuring the comprehensibility of the GNNs is a key element for building a practical GNNs. There are two groups of Comprehensible Graph Neural Networks: one that is interested in Explainable AI, and the other in Comprehensible Machine Learning. The latter focuses on building interpret-able models rather than explaining the output, as is the case in Explainable AI.

2.4.1 Taxonomy

Explainability context can be classified according to the scope of explanation: whether the model can be explained locally or globally, whether the methodology used for explanation is perturbation-based or gradient-based, whether the implementation is model-specific or model-agnostic, and whether it allows for generalization in explainability.

2.4.2 Interpretable GNN

Through sub-graphs

GNNExplainer [17] is a model-agnostic approach that interprets graph neural networks (GNNs) through subgraphs, providing interpretable explanations for the predictions of any GNN-based model. It identifies a compact subgraph structure and a small subset of node features that are crucial to the model's prediction. For explaining a given node's predicted label, GNNExplainer offers a local interpretation by highlighting relevant features and important subgraph structures, identifying the edges most relevant to the prediction. This method was among the first to address explainability in GNNs.

Through Graph Generation

A model-agnostic framework called Graph Neural Networks Including Sparse Interpretability (GISST) [18] interprets important graph structures and node features by discarding unimportant nodes and features through induced sparsity. GISST processes input data to identify important subgraphs and features by estimating the key probabilities in the adjacency and node feature matrices. Additionally, a model-agnostic explainer called Probabilistic Graphical Model for GNNs (PGM-Explainer [19]) identifies crucial graph components to generate explanations. PGM-Explainer produces a simpler, interpretable Bayesian model that illustrates the dependencies among features and provides deeper explanations for GNN predictions.

Through Intermediate Levels Injection

Instead of focusing on subgraphs, interpreting GNNs through graphs generation takes the whole graph structure (or global structure) into consideration. It considers the overall structure of the graph. Then a new graph is generated that contains only the structure necessary for the decision making by GNNs.

Similar to the PGM-Explainer analysing the explained features from conditional probabilities [19], anothermodel-agnostic method of explainable GNNs called PGExplaine [20]. PGExplainer provides explanations for GNNs by generating a probabilistic graph. It is naturally applicable to provide model level explanations for each instance with a global view of the GNN model and has better generalization ability. On the other hand.another also proposed XGNN [21], which provides model-level explanations without preserving the local fidelity. XGNN applied reinforcement learning to generate important graph to explain the prediction

which is made by GNN models. It generates graph patterns by maximizing a certain prediction of the model. Thus it can provide high-level insights and a generic understanding of how GNNs work.

2.5 Applying GNN in RS

In this section, we will discuss the specificity of applying recommendation systems using GNNs, including the challenges and characteristics.

present the different steps mentioned in the previous chapter and what we need to look in each step to build GNN recommendation model.

2.5.1 Constructing the Graph

The first step in applying graph neural networks is to construct the graph. This process involves two key aspects: representing the input data as graph-structured data, and reformulating the recommendation task as a problem on the graph.

As an **example**, consider the standard collaborative filtering task. Here, the input data consists of observed user-item interactions, while the goal is to predict missing user-item interactions. To represent this, a bipartite graph can be constructed with users and items as nodes, and interactions as edges. In this context, the collaborative filtering task becomes a link prediction problem on the graph.

Mathematically, many networks prefer using an adjacency matrix instead of the interaction matrix in collaborative filtering, primarily for computational simplicity.

2.5.2 Network Design

In recommendation systems, GNNs learn embeddings that capture how users interact with items in a graph. These embeddings represent users and items in a vector space, with similar embeddings indicating similar preferences. Some GNNs models also incorporate item and user attributes to enrich the representation.

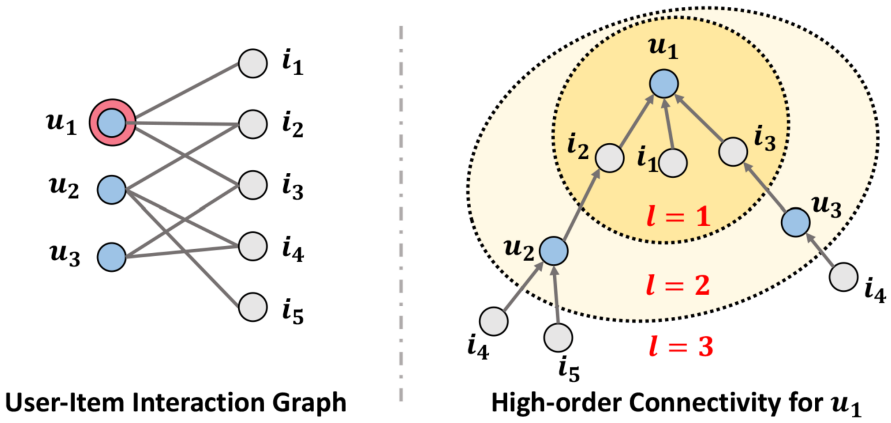

Figure 2.1: High Connectivity

2.5.3 Model Optimization

Various loss functions have been proposed and tailored to suit the unique characteristics of recommendation systems developed with GNNs. These loss functions are designed to address specific challenges, such as handling implicit feedback, or learning embeddings that capture user preferences and item relevance accurately. while some other are optimized for evaluations concepts like explainability.
Furthermore, GNNs-based recommendation systems may encompass multiple tasks, such as link prediction tasks across diverse types of edges. In such scenarios, striking a balance between these tasks and leveraging their mutual reinforcement poses a formidable challenge.

Cross-Entropy Loss

The cross-entropy loss is commonly used for classification tasks in GNN-based recommender systems. It is formulated as 2.7

$$L = -\sum_{(i,y_i)\in O} y_i^T \log p_i \tag{2.7}$$

where y_i is the ground truth label for user-item pair i, and p_i is the predicted probability. This loss function aims to minimize the difference between the predicted and true labels.

For example, in a binary classification task where we predict whether a user will like or dislike an item, the ground truth label y_i could be 1 if the user likes the item and 0 otherwise. If the model predicts a probability p_i of 0.8 for the user liking the item, the cross-entropy loss for this example would be $L = -(1 \times \log(0.8) + 0 \times \log(1 - 0.8))$.

Pairwise Loss

Pairwise loss functions, such as Bayesian Personalized Ranking (BPR) loss, are widely used in GNN-based recommender systems. The idea is to learn a ranking function that can correctly order the relevant and non-relevant items for each user. The BPR loss is defined as **??**

$$L = -\sum_{(i,j)\in O} \log \sigma(x_i - x_j) \tag{2.8}$$

where x_i and x_j are the predicted scores for the positive and negative samples, respectively, and σ is the sigmoid function.

Point-wise Loss

Point-wise loss functions, like Mean Squared Error (MSE) or log-likelihood, are used when the goal is to predict the exact rating or preference score for each user-item interaction. The point-wise loss aims to minimize the difference between the predicted and true values for each observed interaction.

For this example, In a scenario where we are predicting the rating a user would give to an item on a scale of 1 to 5, and the ground truth rating for a user-item interaction is 4. If the model predicts a rating of 3.5 for this interaction, the point-wise loss (e.g., Mean Squared Error) would be $(4 - 3.5)^2$.

Adversarial Loss

Adversarial training can be incorporated into GNN-based recommender systems to improve robustness and generalization. The adversarial loss encourages the model to learn representations that are invariant to small perturbations of the input, helping to mitigate the impact of noisy or sparse data.

Multi-Task Loss

For recommender systems with multiple objectives, such as accuracy, diversity, and fairness, a multi-task loss function can be used to optimize the model for all the desired goals simultaneously. This allows the GNN-based model to learn representations that balance the different recommendation requirements.

2.6 Models Review

2.6.1 Graph Convolution Network

Graph Convolution Network (GCN) is one of the earliest works in GNNs. Neural Graph Collaborative Filtering is a GCN variant that uses the user-item interactions to learn the collaborative signal, which reveals behavioral similarity between users, to improve recommendations. Rating predictions on Yelp2018 and Amazon-book datasets were used to measure the performance of the model.

In Table 2.2, different Graph Convolutional Network model variants developed for various tasks are presented. We will later detail the most promising versions, including Neural Graph Collaborative Filtering and the Light Graph Convolutional Network.

Table 2.2: Collaborative Filtering GNN

Name	Paper	Year
GCMC	Graph convolutional matrix completion.	2017
Pin-Sage	Graph convolutional neural networks for web-scale recommender systems.	2018
NGCF	Neural graph collaborative filtering.	2019
LightGCN	Lightgcn: Simplifying and powering graph convolution network for recommendation.	2020
NIA-GCN	Neighbor interaction aware graph convolution networks for recommendation.	2020
DGCF	Disentangled graph collaborative filtering.	2020
IMP-GCN	Interest-aware message-passing gcn for recommendation.	2021
SGL	Self-supervised graph learning for recommendation.	2021
LT-OCF	LT-OCF: Learnable-Time ODE-based Collaborative Filtering.	2021
HMLET	Linear, or Non-Linear, That is the Question!	2022
HS-GCN	HS-GCN: Hamming Spatial Graph Convolutional Networks for Recommendation.	2022
LGCN	Low-pass Graph Convolutional Network for Recommendation.	2022

Neural Graph Collaborative Filtering

At its core, Neural Graph Collaborative Filterin (NGCF) employs a neural network architecture that learns low-dimensional representations (embeddings) of users and items by aggregating information from their neighboring nodes in the interaction graph. Unlike traditional collaborative filtering methods that rely solely on user-item interactions or traditional embeddings that embed only the item or user attributes independently, NGCF considers the entire graph structure, allowing it to capture higher-order connectivity and dependencies between users and items [22] .

The following figure 2.2 demonstrate the architecture, by first building the *embedding* E of both the users and the items, then the message passing architecture of graph neural network is constructed in the *embedding propagation* layers L, than *the prediction layer* that take different embedding representation of single item and user and predict the link between them.

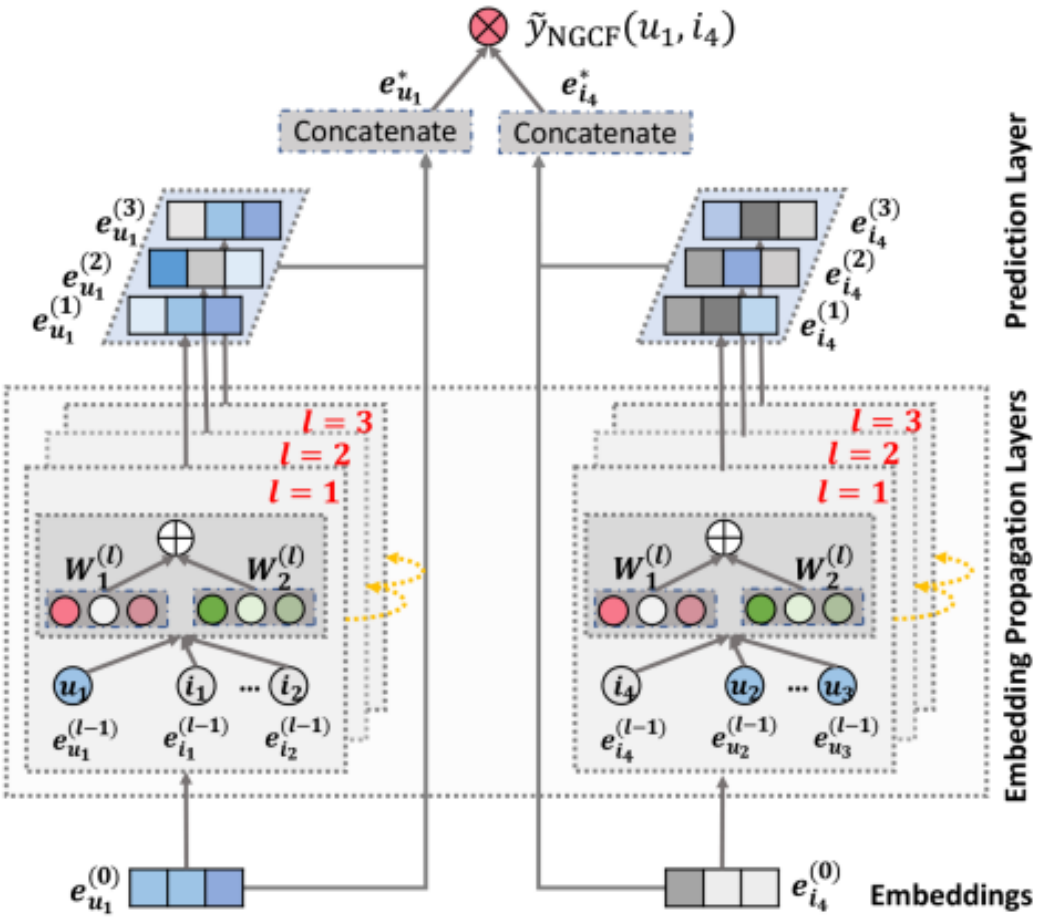

Figure 2.2: An illustration of NGCF model architecture

Light Graph Convolutional Network

Light Graph Convolutional Network (LightGCN) is described a simplification of NGCF that profit from the aggregation. In the aggregation function the (LightGCN) architecture uses simple weighted sum aggregator rather than the use of feature transformation and nonlinear activation of different aggregations based [23].

2.6.2 Graph Attention Network

Graph Attention Network [24] uses an attention mechanism [25] to learn the influence of neighbors; this influence is used to determine the contribution of neighbors during the aggregation step.
Multi-Component Graph Convolutional Collaborative Filtering (MCCF) [26] is one such approach that learns latent purchasing motivation using an attention mechanism and combines it with features from explicit user-item interactions for better recommendations. The authors benchmarked MCCF's performance on MovieLens, Amazon Product Recommendation, and Yelp datasets.

2.6.3 GraphSage

GraphSage is a framework that proposes sampling fixed-sized neighborhoods instead of using all the neighbors of each node for aggregation [27]. It also provides min, max, or sum pooling as options for aggregators and uses concatenation operation to update node/edge/graph representations.
PinSage [28] is a variant of GraphSage proposed as a solution to handle web-scale graphs. PinSage, from Pinterest, introduces a sampling technique that samples fixed-size neighborhoods based on the highest visit counts. Alternatively, visit counts can be substituted with any feature that differentiates the node's importance.

2.7 Conclusion

In conclusion, Graph Neural Networks (GNNs) have proven to be versatile tools for tackling graph-structured data, offering broad applications in fields like biology, chemistry, and recommendation systems. Throughout this chapter, we explored the design pipeline for GNNs, from graph construction to model training, focusing on different graph types and their roles in optimizing model performance. The implementation of advanced techniques such as message passing, loss functions like Bayesian Personalized Ranking (BPR), and sampling methods further demonstrates the potential of GNNs to enhance computational efficiency and accuracy in complex tasks. As we examined models like Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs), the capacity of GNNs to generalize traditional neural network architectures for graph data became clear. Ultimately, GNNs stand as powerful, adaptable tools for diverse applications, especially when optimized for specific graph structures and tasks.

Chapter 3

Neuro-Symbolic Recommendation System

3.1 Introduction

Neuro-Symbolic reasoning refers to a form of reasoning that operates on symbols and logical rules to derive new knowledge from existing knowledge. In symbolic reasoning, information is represented using symbols, and logical operations are applied to manipulate and infer relationships between these symbols. This form of reasoning is often associated with classical logic and symbolic AI (artificial intelligence) systems. Conversely Neural Network are computational form of reasoning that uses interconnected nodes defined as neurons where these neurons receive input, process it through weighted connections, and produce output.

In literature the integration of the symbolic reasoning into neural reasoning is promising field to overcome a lot of mathematical and information representation limitations, specially in complex systems of recommendations, where their decision process is getting more difficult specially with use of novel neural graph architecture in RS.

In this chapter we will see how different neural symbolic are integrated at different approach concept to handle some scenarios and evaluation concepts. In the first section we well describe the different models where give for each model a review of it's position in relevance with the recommendation application and scenarios.

3.2 Neuro-Symbolic Application

Neuro-symbolic systems combine the strengths of symbolic reasoning and neural networks, addressing challenges in artificial intelligence (AI) that neither approach can solve independently. Traditional symbolic AI, with its rule-based logic, excels at tasks requiring explicit reasoning and explainability, while neural networks, particularly deep learning models, are adept at handling raw data, pattern recognition, and generalization from examples. Neuro-symbolic systems bridge these approaches, enabling the integration of high-level

reasoning with low-level perception.

Applications of NeSy systems are diverse, spanning fields like natural language processing, computer vision, robotics, and even game playing. For instance, in natural language processing, NeSy models can interpret complex linguistic structures by combining symbolic grammar rules with neural models for word embeddings. In robotics, these systems can enable robots to perform tasks by understanding both the physical environment (via neural perception) and abstract instructions (via symbolic reasoning). The combination of symbolic and neural methods leads to more robust, interpretable, and flexible AI systems capable of performing complex tasks in dynamic environments.

As seen in Figure 3.1, symbolic reasoning, often called symbolic AI, consists of two major components: the knowledge base and the inference engine. The knowledge base contains all the defined rules, while the inference engine takes specific questions or queries from user input and processes them through the available rules in the knowledge base to produce an output.

In contrast, deep neural networks work quite differently. Initially, they do not hold any predefined rules but start with randomly initialized parameters. These parameters are adjusted during training using historical data, allowing the model to gradually learn patterns and implicit "rules" that can be applied to new inputs to generate outputs. This training process constructs a set of decision-making mechanisms that, although not explicitly rule-based, enable the network to respond effectively to future questions.

When it comes to merging the two approaches, there are several advantages to be listed:

3.2.1 Advantages of Neuro-Symbolic Methods

Neuro-symbolic methods combine the strengths of symbolic reasoning and deep learning, offering several benefits:

- Improved Interpretability: By combining symbolic reasoning with neural networks, neuro-symbolic methods provide a level of interpretability that purely neural approaches often lack. Symbolic reasoning allows for rule-based explanations, making it easier to understand why a decision was made.
- **Knowledge Transfer**: Symbolic knowledge can be reused across different tasks, enabling more efficient knowledge transfer. Instead of retraining a model from scratch, existing knowledge from symbolic reasoning can be leveraged to reduce the training time for new tasks.
- **Better Generalization**: The use of symbolic rules in neuro-symbolic systems allows them to generalize better, especially in scenarios with limited data. Symbolic rules help guide the network by providing structure, which is particularly beneficial when data is sparse or noisy.
- **Handling Complex Logical Relationships**: Symbolic methods excel at capturing complex logical relationships, while deep neural networks are effective at extracting abstract patterns from raw data. Merging these two capabilities allows neuro-symbolic models to handle both the structured logic of symbols and the unstructured data handled by neural networks.

- **Overcoming Data Limitations**: Purely data-driven methods like deep neural networks require a large amount of training data to learn effectively. Neuro-symbolic methods can leverage symbolic knowledge bases to reduce the dependence on large datasets, thereby enhancing the model's performance even with fewer data samples.
- **Robustness and Flexibility**: Neuro-symbolic methods can lead to more robust systems that are less prone to the pitfalls of pure deep learning models, such as overfitting. The symbolic component provides a rule-based mechanism that can act as a safeguard in situations where the neural component may produce uncertain or incorrect outputs.
- **Improved Reasoning Capabilities**: Neuro-symbolic systems are capable of **both learning from data and reasoning with explicit rules**, making them uniquely positioned to perform complex reasoning tasks. This dual capability makes them especially useful in areas requiring a high level of understanding and reasoning, such as decision support systems, scientific discovery, and complex games.

These advantages highlight why neuro-symbolic approaches are a promising avenue in the development of recommendation systems and other AI applications that benefit from both interpretability and the capability to learn from large datasets. By combining symbolic knowledge with the adaptability of deep neural networks, we can create systems that are not only powerful and accurate but also capable of providing clear, interpretable reasoning behind their decisions.

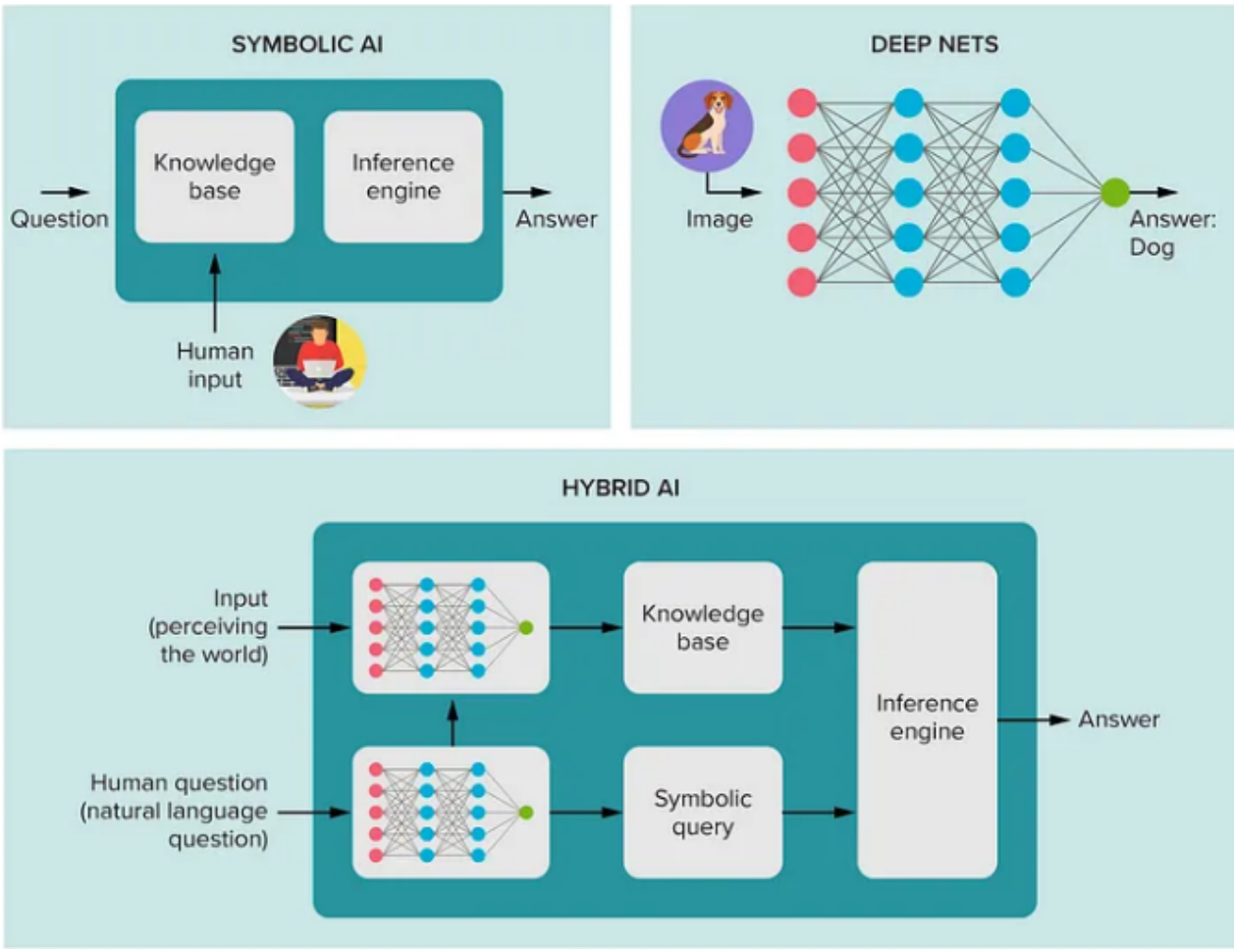

Figure 3.1: Illustration of Neural and Symbolic Reasoning

3.3 Technics to Develop Neuro-Symbolic Models

Developing NeSy models requires a deep understanding of both symbolic reasoning and neural network architectures. One common technique is to create hybrid models where neural networks handle perception and data interpretation, while symbolic AI components manage high-level reasoning and decision-making. This can be achieved by integrating logic programming languages like Prolog with neural networks, enabling the system to reason about the data processed by the neural component.

Another technique involves the use of embeddings that map symbolic knowledge into a continuous space where it can be processed by neural networks. For example, Knowledge Graph Embeddings (KGEs) are used to represent entities and relations in a knowledge graph as vectors, which can then be fed into a neural network. This approach allows symbolic knowledge to be utilized in deep learning models, facilitating tasks like relation extraction and reasoning over knowledge bases.

Transfer learning can also be leveraged in neuro-symbolic systems, where a pre-trained neural network (trained on a large dataset) is fine-tuned with symbolic constraints to improve performance on specific tasks. Additionally, reinforcement learning can be combined with symbolic reasoning to guide the training of neural networks through symbolic goals or constraints, leading to more efficient learning processes.

3.4 Applying Neuro-Symbolic into RS

Applying Neuro-Symbolic (NeSy) methods into Recommendation System represents a cutting-edge approach to improving recommendation accuracy, interpretability, and robustness. Traditional recommender systems, often based on collaborative filtering or content-based methods, struggle with explainability and often require large datasets to perform well. Neuro-symbolic approaches can address these limitations by incorporating symbolic reasoning to better understand user preferences and make more informed recommendations.

For example, in a movie recommender system, a neuro-symbolic approach could combine neural networks to analyze user viewing patterns with symbolic reasoning to understand genres, actors, and directors the user prefers. This enables the system to make recommendations that are not only based on past behavior but also align with explicit user preferences or constraints, making the recommendations more interpretable.

Additionally, by integrating knowledge graphs with recommender systems, neuro-symbolic methods can provide richer contextual understanding. For instance, a recommender system could use a knowledge graph to understand relationships between different products, enabling it to suggest complementary items (e.g., recommending a charger when a user views a smartphone). By applying neuro-symbolic methods, recommender systems can achieve higher levels of personalization, explainability, and user satisfaction.

3.5 Models Review

3.5.1 Neural Collaborative Reasoning

Their researches were limited only to the scenario of sequence recommendation, the competitors being the **Neural Collaborative Reasoning**(NCR). that only formalizes the sequential recommendation scenario as a logical reasoning problem.but doesn't formalize the end to end decision process of the recommended system example of this is to recommend the user item 4 because he chose the item 3 and 1.

3.5.2 Graph Collaborative Reasoning

Integration of GNN into the NCR were also made but suffer from the same weak explanation of decision process.the Counter Collaborative Reasoning which is also based on NCR.

3.5.3 Counter Factual Reasoning

counterfactual logic reasoning is exploited to generate counterfactual examples for data augmentation based on NCR The examples are generated by discovering slight changes in users' explicit feedback (i.e., the sequence of purchases) by solving a counterfactual optimization problem. framework can generate explicit counterfactual explanations to understand the user behavior sequence

3.5.4 HYbrid Probabilistic Extensible Recommended

Unlike the previous reasoning integration that only focus on small part of sequence recommendation, which is based on Probabilistic Soft Logic.

In particular, HYbrid Probabilistic Extensible Recommended (HyPER) exploits the First-Order Logic (FOL) to encode knowledge from a wide range of information sources, such as multiple user and item similarity measures, content, and social information. For example, the FOL formula is used to express that if users 1 and 2 are similar according to similarity measure and 1 likes item then 2 should also like Similar formulas are used to express other kinds of facts. Then, Hinge-Loss Markov Random Fields are used to learn how to balance the different information types. The main concern of HyPER is scalability due to the usage of Markov Logic Networks. HyPER is highly related to the Logic Tensor Network since the logical formulas resemble each-other.

3.5.5 Integration Symbolic into Graph Embedding

proposed using a NeSy approach to encode FOL formulas to enhance knowledge graph embeddings and provide accurate knowledge-aware recommendations. Their approach consists of three steps:

1. the FOL formulas are automatically extracted from a recommendation knowledge graph.

2. the knowledge graph embedding are learned jointly with the extracted formulas using a NeSy approach.

3. Finally, the user-item embedding are fed to a neural architecture to get predictions.

3.5.6 Logic Tensor Neural

Like HyPER, the Logic Tensor Neural LTN, use First Order Logic to train a vanilla Matrix Factorization model using a First-Order Logic knowledge base as an objective. In particular, we encoded facts to enable the regularization of the latent factors using content information, obtaining promising results. they proposed different directions how can their approach can overcome limitation in cross domain scenario recommendation and explainable without actual performing it.

3.5.7 Multi layered Logical Perceptron

The rule-based model [29], built on a neural network architecture with a logical activation function, is primarily designed for classification tasks. the recommendation can be easily adapted to classification by restructuring the task. To enhance the model's ability to distinguish between complex attribute interactions, a novel training method known as Random Binarization is introduced. This method works by randomly selecting a subset of the model's weights and binarizing them during the training process. The binarization forces the network to learn discrete, rule-based decisions, thereby enabling the Multi Logical Layer Perceptron to capture the relationships between various attributes that are crucial for effective classification. However, a notable limitation of the MLLP is that it is specifically optimized for datasets with discrete attributes, which may reduce its generalization capability when working with continuous or mixed data types.

3.5.8 Rule Representation Learning

Another rule based model is Rule Representation Learning (RRL) [30] which can automatically learn interpretable rules for data representation and classification the same as MLLP,where it can be applied for recommendation by changing the recommendation task into classification task. the particularity of RRL, is a new gradient-based discrete model training method, i.e., Gradient Grafting, that directly optimizes the discrete model. it also propose an improved design of logical activation functions to increase the scalability of RRL and make RRL capable of discretizing the continuous features end-to-end. it experimental results show that RRL has both high classification performance and low model complexity on data sets with different scales.

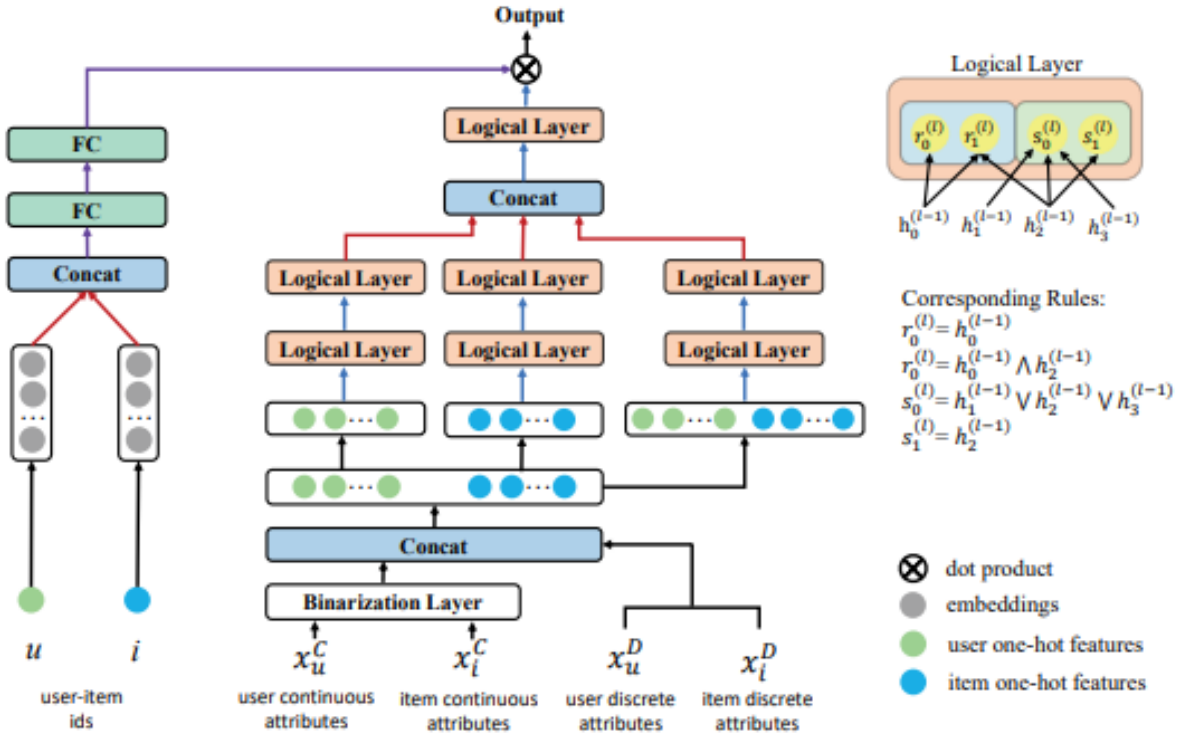

Figure 3.2: NSICF Model Architecture

3.5.9 Neuro-Symbolic Interpretable Collaborative Filtering for Attribute-based Recommendation

Neuro-Symbolic Interpretable Collaborative (NS-ICF) as shown in 3.2 learns Interpretable recommendation rules (consisting of user and item attributes) based on neural networks with two innovations: (1) a three-tower architecture tailored for the user and item sides in the RS domain; (2) fusing the powerful personalized representations of users and items to achieve adaptive rule weights and without sacrificing Interpretablity. Comprehensive experiments on public datasets demonstrate NS-ICF is comparable to state-of-the-art deep recommendation models and is transparent for its unique neuro-symbolic architecture [29].

3.6 Comparative Analysis of Neuro-Symbolic Approaches

In this section, we compare the strengths and limitations of the different neuro-symbolic approaches discussed previously. The following tables 3.1 3.2 summarize these comparisons.

Table 3.1: Strengths of Neuro-Symbolic Approaches in Recommendation Systems

Approach	**Strengths**
Neural Collaborative Reasoning (NCR)	Effective in sequential recommendations; introduces logical reasoning into neural models.
Graph Collaborative Reasoning	Enhances handling of relational data; integrates GNNs into recommendation models.
Counterfactual Reasoning	Provides explicit counterfactual explanations; improves understanding of user behavior.
Hybrid Probabilistic Extensible Recommendation (HyPER)	Combines multiple information sources; uses FOL for encoding knowledge; balances different data types effectively.
Integration of Symbolic Logic into Graph Embedding	Enhances knowledge-aware recommendations; combines symbolic logic with graph embeddings.
Neuro-Symbolic Interpretable Collaborative Filtering (NS-ICF)	Maintains interpretability while achieving strong performance; introduces innovative architecture tailored for recommendation systems.
Logic Tensor Networks (LTN)	Regularizes latent factors with logical constraints; offers potential solutions for cross-domain scenarios.

3.7 Conclusion

The integration of neuro-symbolic methods into recommendation systems represents a significant leap forward in both the accuracy and interpretability of these systems. By combining the high-level reasoning capabilities of symbolic AI with the adaptability and pattern recognition strengths of neural networks, neuro-symbolic systems are able to overcome some of the limitations faced by traditional recommender systems. This hybrid approach offers a pathway to more explainable, scalable, and context-aware recommendations.

Throughout this chapter, we have examined various models and techniques that utilize neuro-symbolic reasoning in different recommendation scenarios. These models, ranging from neural collaborative reasoning to logic tensor networks, demonstrate the potential of neuro-symbolic systems to handle complex user behaviors and diverse information sources. Moreover, the application of first-order logic, knowledge graph embeddings, and counterfactual reasoning has shown promise in addressing the challenges of scalability, decision transparency, and user satisfaction.

While these advancements are promising, several challenges remain, such as improving the scalability of more complex symbolic models and refining the integration of symbolic

Table 3.2: Limitations of Neuro-Symbolic Approaches in Recommendation Systems

Approach	**Limitations**
Neural Collaborative Reasoning (NCR)	Limited to sequential recommendations; lacks comprehensive decision process explanation.
Graph Collaborative Reasoning	Still suffers from weak explainability despite GNN integration; scalability concerns with large graphs.
Counterfactual Reasoning	Dependent on quality of counterfactual examples; computational complexity in generating examples.
Hybrid Probabilistic Extensible Recommendation (HyPER)	Scalability issues due to reliance on Markov Logic Networks; complex model requires careful tuning.
Integration of Symbolic Logic into Graph Embedding	Complexity in extracting and integrating FOL formulas; potential scalability challenges.
Neuro-Symbolic Interpretable Collaborative Filtering (NS-ICF)	Potentially higher computational cost due to three-tower architecture; requires balanced fusion of user-item representations.
Logic Tensor Networks (LTN)	Mostly theoretical applications; lacks empirical evidence in large-scale scenarios.

reasoning into neural architectures. Nonetheless, neuro-symbolic systems are positioned to play an increasingly important role in the future of recommendation systems, offering a bridge between human-like reasoning and powerful data-driven insights. Moving forward, research in this field will likely focus on enhancing the efficiency of these systems while preserving the balance between interpretability and performance.

Part I

Model Development Methodology

Chapter 4

Proposed Neuro-Symbolic Collaborative Filtering Based Explanation Model

4.1 Introduction

To tackle the mentioned challenges to ensure accurate and explained recommendation. Neuro-Symbolic Collaborative Filtering Based Explanation ModeL NSXCF model represents a significant advancement by integrating neural network methodologies with symbolic reasoning to enhance both recommendation accuracy and explainability. our model incorporate two different explanation style, the first by handling the attributes of items and users differently and offering generalizing rules and secondly with user and item it will perform collaborative filleting and uses the similarity for provide neighboring items.

In this chapter, we present the NSXCF model in detail. In the first section, We begin with an overview of the model's architecture, describing how it incorporates user and item attributes and employs collaborative filtering techniques. The second section delve into the specifics of attribute explanation styles, including the application of logical rules to attribute binarization and rule construction. We then in the third section explore the methodology for generating similar item explanations through collaborative filtering. Finally, in the last section we discuss the model's training process, including the use of Binary Cross-Entropy Loss (BCELoss) and sampling strategies for optimization.

4.2 Model Overview

The Neuro-Symbolic Collaborative Filtering NSXCF model integrates symbolic reasoning with neural collaborative filtering to provide accurate and interpretable recommendations.

Figure 4.1 illustrates the architecture of the NSXCF model. The model processes User ID U and Item ID I inputs, along with their respective attributes: User Attributes A_u and Item Attributes A_i. These attributes are binarized, and rule attribute weights W_A are applied to create a weight matrix where A denotes the number of attributes and D represents the number of logical conjunction rules constructed.

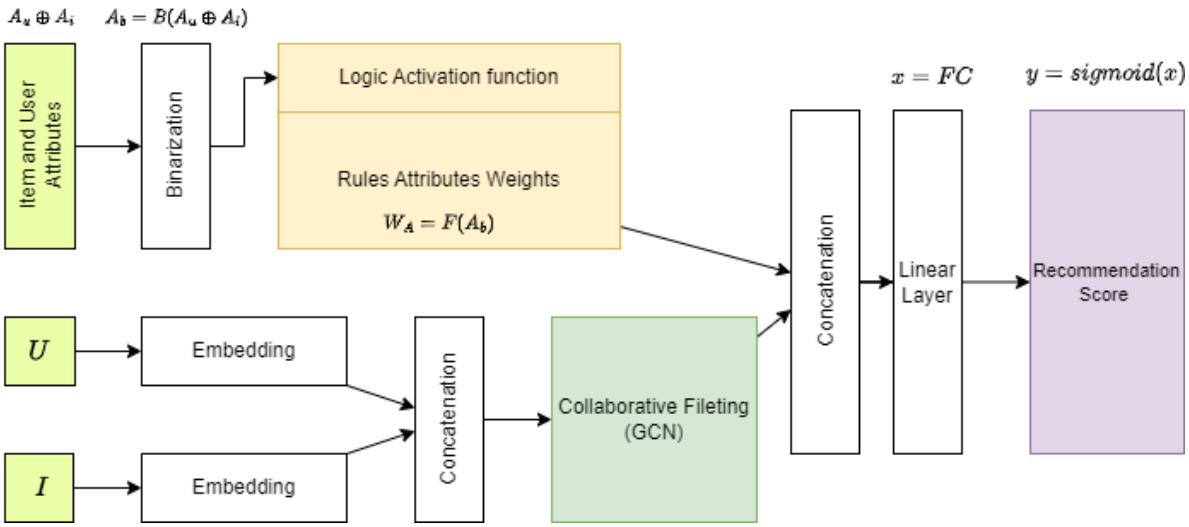

Figure 4.1: Proposed Model Architecture

Embeddings for User ID and Item ID are computed and updated through collaborative filtering. The concatenated outputs from these layers are passed through a linear layer with a sigmoid activation function to produce the recommendation score, which is bounded between 0 and 1. The model outputs include the recommendation score, attribute rules derived from the logical conjunction layer, and similar items identified through collaborative filtering.

4.3 Attributes Explanation Style

Inspired by the logic-based methods from the Transparent Classification with Multilayer Logical Perceptrons (MLLP) model [29], our model applies a logical activation function to generate interpretable attribute rules. After binarizing both User Attributes (A_u) and Item Attributes (A_i), these vectors are combined into a single binarized vector (A_b). This vector is then used in a logical conjunction function to construct the weight matrix W_A, where each row represents a logical rule, and each column corresponds to a binarized attribute.

The logical conjunction output is computed as 4.1:

$$O_{\text{conj}}(x) = \bigwedge_{i=1}^{n} F_c(x_i, m_i) \tag{4.1}$$

Where:

$$F_c(x_i, m_i) = x_i m_i = 1 - m_i(1 - x_i) \tag{4.2}$$

The parameters are:

- $x_n \in 0, 1$: Input binarized vector.
- n: Number of inputs in the logical neuron.

- $F_c(x_i, m_i)$: Boolean function for each input.
- O_{conj}: Output of the conjunction neuron.
- m_i: Weight parameter defined as $m_i = \text{Sigmoid}(cw_i)$, where $c \geq 1$ is a constant.

After training, the model selects the top-k attribute combinations (e.g., top 5, 10, or 15 attributes) for each rule. Attributes that are exclusive (e.g., gender) are treated separately, with the most significant attribute being selected.

4.4 Similar Item Explanation Style

In addition to generating attribute-based explanations, our model incorporates collaborative filtering techniques to identify and recommend similar items. This process involves embedding both item and user IDs into a latent space where the relationships between them are captured. By applying these embeddings to a bipartite graph—a type of graph that connects users with items—the model can explore and analyze the interactions between users and items.

The bipartite graph is represented through an adjacency matrix, which captures the connections between users and items. This matrix is then used to construct final embeddings through the state-of-the-art collaborative filtering approach known as LightGCN (Light Graph Convolution Network) [23]. LightGCN is renowned for its ability to achieve optimal recommendation accuracy by simplifying and enhancing traditional graph convolutional networks. It leverages neighborhood information effectively, allowing for more precise and efficient computation of item similarities.

These final embeddings facilitate the identification of items that are similar to those the user has previously interacted with. By integrating the collaborative filtering results with the attribute-based explanations, the model provides a comprehensive explanation for the recommendations. This dual approach ensures that users receive not only insights into why specific items were recommended based on their attributes but also how these items relate to their previous interactions. This multi-faceted explanation enriches the user experience by offering a clearer rationale behind each recommendation and improves overall recommendation accuracy by leveraging both attribute information and interaction patterns. The approach used to find and explain similar items. After embedding User ID and Item ID, collaborative filtering is applied using an adjacency matrix derived from bipartite interactions. This matrix helps identify neighboring items, ensuring that recommendations are aligned with previous user interactions.

4.5 Model Training

The training of the NSXCF model involves the optimization of embeddings and rule weights using Binary Cross-Entropy Loss (BCELoss). Positive and negative samples are used for training, where positive samples are user-item pairs with interactions and negative samples are those without interactions.

The BCELoss function is defined as:

$$\text{BCELoss} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i^+ \log(p_i^+) + (1 - y_i^-)\log(1 - p_i^-)\right] \tag{4.3}$$

where:

- N is the total number of samples.
- y_i^+ is the actual label for positive samples (usually 1).
- y_i^- is the actual label for negative samples (usually 0).
- p_i^+ is the predicted probability for positive samples.
- p_i^- is the predicted probability for negative samples.

The model updates its parameters to maximize the probability of correct recommendations and minimize the probability of incorrect ones, ensuring effective learning and accurate predictions.

4.6 Example Interpretation

In ordered to understand more about our proposed architecture we provide this example that detail how the input can construct our expected output and different reasoning, The example sketched in Figure 4.2 provides an overview of NSXCF for a user U_1, who already watched three movies $I_1, I_2 and I_3$. The model provides three outputs:

- A recommendation score for the specific item I_5 with a score of 0.85 through a multi-layered architecture.
- Similar item(s) to the recommended item I_5 are also provided thanks to the collaborative filtering process. In this example, it's the item I_3 with the similarity score of 0.7.
- User and item attributes A_u1 and $A_i1+A_i2+A_i3$ that match the conjunction attributes rule :
  Movie_year $= 1998 \wedge$ Occupation $=$ Unemployed $\wedge$ Age < 20.

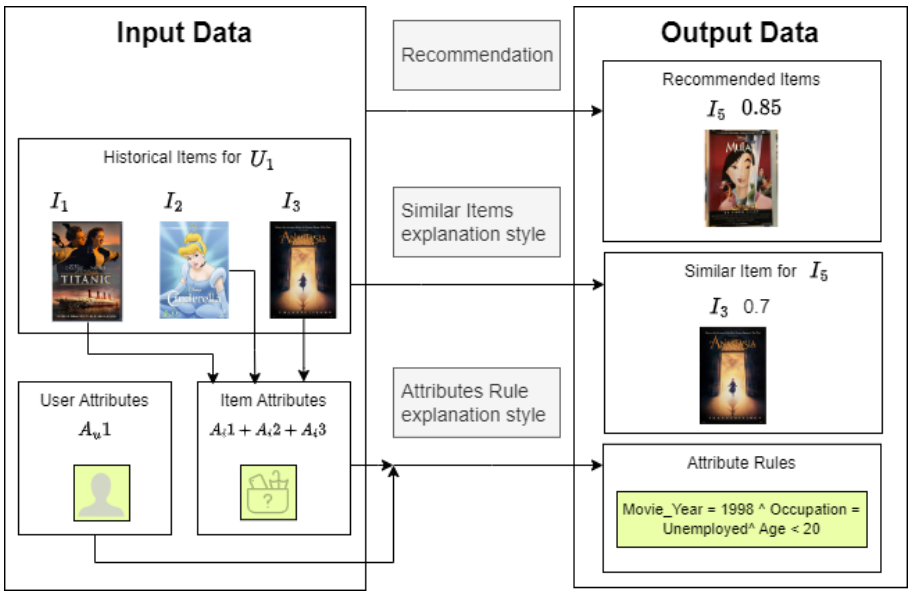

Figure 4.2: The Model input and output example interpretation

4.7 Conclusion

This chapter presented the NSXCF model, highlighting its novel integration of symbolic reasoning with neural collaborative filtering. The model enhances recommendation accuracy and interpretability through detailed attribute rules and similar item explanations. Future work may focus on improving scalability and exploring additional interpretability features.

Chapter 5

Experimentation

5.1 Introduction

In this chapter, we present the experimental setup and results of our evaluation of the Neuro-Symbolic Collaborative Filtering (NSXCF) based explanation model. The evaluation aims to assess the effectiveness of the NSXCF model across various recommendation environments by utilizing two distinct datasets, ML-100k and Taobao. We explore both the technical setup and the performance outcomes of our model, offering a comprehensive view of its capabilities and limitations.

The chapter is structured as follows:

- First, we discuss the datasets chosen for the experiments, highlighting their unique properties and why they are appropriate for evaluating the NSXCF model.
- Next, we detail the evaluation metrics used to measure the model's performance, including recommendation scores, attribute rule explanations, and similar item recommendations.
- We then present the results of the experiments, comparing the NSXCF model against state-of-the-art techniques in recommendation tasks.
- This is followed by an ablation study that examines the contributions of different model components to overall performance.
- Lastly, we analyze the effectiveness of the Attribute Rule and Similar Item Explanation Styles, and conclude with a summary of findings and implications for future research.

By merging the experimental setup and results into one coherent narrative, this chapter offers a holistic view of the NSXCF model's performance across multiple dimensions.

5.2 Datasets

To conduct a robust evaluation of the NSXCF model, we selected two datasets that represent different types of recommendation environments: the Taobao dataset [31] and the

MovieLens (ML-100K) dataset [32]. These datasets were chosen based on their density and diversity, allowing us to compare the model's performance under varying conditions. A detailed explanation of the properties of these datasets is given below:

- **Taobao Dataset:** The Taobao dataset is characterized by a large number of users and items, but relatively few interactions between them, leading to a highly sparse matrix. This makes it representative of real-world e-commerce platforms, where users tend to browse many items but interact with only a small fraction. Handling such sparsity is a common challenge in recommendation systems, and this dataset allows us to evaluate the model's ability to manage low-density data effectively. For instance, recommending the right products to a new user who has only interacted with a few items is crucial for a successful e-commerce business, which makes Taobao a valuable dataset for testing.
- **MovieLens (ML-100K) Dataset:** This dataset, in contrast, has a smaller number of users and items but features significantly more interactions. The ML-100K dataset is high-density, which makes it suitable for testing models in environments rich with user-item interaction data. Dense datasets are ideal for validating how well a model can learn complex user preferences. The ML-100K dataset, which contains movie ratings and metadata such as genre and user demographics, also allows us to test the integration of such attributes in model explanations, leading to more personalized recommendations.

The Table 5.1 summarizes the key characteristics of the datasets used in our experiments.

Table 5.1: Datasets Descriptions

Dataset	N° Users	N° Items	N° Interactions	N° User Attributes	N° Item Attributes
ML-100k	5000	2000	100000	24	17
Taobao	15900	6249	67903	4	1947

The diversity between Taobao and ML-100k datasets also provides an opportunity to observe how the NSXCF model handles cold-start problems in recommendation. In the case of Taobao, which exhibits a lower density, cold-start scenarios are more common, such as when new items are added to the system or new users register with minimal interaction history. ML-100K, with its rich user-item interaction matrix, helps us analyze the impact of dense data on model behavior.

5.3 Evaluation Metrics

The evaluation of the NSXCF model is performed across its three main outputs: the recommendation score, attribute rules, and similar items. Each output is assessed using specific metrics tailored to its nature. Evaluation metrics are crucial for understanding different aspects of the recommendation model's performance, including accuracy, coverage, novelty, and explainability.

- ***Recommendation Score Evaluation:*** is measured using the Normalized Discounted Cumulative Gain (NDCG@K) [33]. NDCG is used because it effectively captures the

relevance of ranked items while emphasizing the importance of position within the recommendation list. For example, if a user prefers a particular genre of movies, NDCG helps evaluate if those genres are placed in the top of the recommended list, hence increasing user satisfaction. By focusing on the top-K items, this metric helps us evaluate the quality of rankings from the user's perspective, with K set to either 10 or 20 depending on the dataset's density of recommendation interactions. This ensures we assess whether the most relevant items are ranked higher, which is crucial in personalized systems where users are likely to focus on top results. Additionally, we use the Area Under the Receiver Operating Characteristics Curve (ROC-AUC) to evaluate the overall accuracy of the recommendations. ROC-AUC provides a balanced view of the model's performance by considering both true positive and false positive rates. It is suitable for evaluating binary relevance, where it is important to assess how well the system can differentiate between relevant and non-relevant items across different thresholds.

- ***Attribute Rule Evaluation:*** evaluates the alignment of generated attribute rules with the final recommended items. We specifically consider the average score of matching between attribute rules and the recommended items. A higher matching score indicates that the model can effectively utilize user and item attributes to make tailored recommendations, which adds to its interpretability. This metric helps to understand if the generated explanations reflect the real user preferences, thereby bridging the gap between user understanding and model decisions.

- ***Similar Items Evaluation:*** is used to assess the model's ability to recommend items that align with a user's previously shown preferences. To evaluate this, we calculate the proportion of previously selected items that are similar to the newly recommended items using cosine similarity. This metric allows us to determine if the model can accurately identify and suggest items with similar features, which can enhance user satisfaction by focusing on consistency in preferences. For instance, a user who frequently interacts with action films would expect the system to recommend films similar in theme and genre, which improves their overall experience.

Overall, these metrics provide a comprehensive assessment of the model's capability to generate useful, accurate, and explainable recommendations.

5.4 Competitors

To benchmark the NSXCF model, we selected several competing models that share a focus on explainable AI and have been previously tested on similar datasets and evaluation metrics. These models include:

- RRL: This neuro-symbolic architecture is primarily designed for classification tasks with model-level explainability. By tackling the recommendation task as a classification problem, we apply the RRL model to both datasets to evaluate its performance in this context. Given its neuro-symbolic architecture, the RRL model is particularly strong

in generating symbolic explanations that align with user preferences, which allows a comparative analysis of symbolic reasoning versus hybrid reasoning in recommendation.

- XGBoost: A tree-based ensemble learning model, XGBoost is often regarded as a complex model with inherent explainability features. This model is efficient in capturing non-linear relationships between features. In the context of recommendation tasks, XGBoost's tree-based explainability gives insights into which features have a higher influence on the final prediction, making it suitable for comparison with our attribute-based explainable model.

- NSICF: This model specializes in attribute-based recommendations, exclusively utilizing attribute rules within its architecture. It is the most similar to NSXCF in terms of design, as it transforms the RRL architecture into one that can handle recommendation tasks. However, NSICF is limited to providing redundant attribute rules, whereas NSXCF not only generates unique attribute rules but also offers additional explanations for aspects that NSICF does not address. For instance, NSXCF extends beyond attribute rules to provide complementary insights such as similar item recommendations, enhancing the comprehensiveness of explanations.

These competitor models serve as important benchmarks, helping us evaluate both recommendation quality and explainability compared to the hybrid neuro-symbolic reasoning capabilities of the NSXCF model.

5.5 Recommendation Analysis

In this section, we compare the recommendation performance of the NSXCF model with state-of-the-art models across the ML-100k and Taobao datasets. Our results reveal that the NSXCF model excels in ranking metrics, particularly for the ML-100k dataset. This dataset is characterized by a smaller number of users and items but a higher number of interactions, making it a dense dataset. The superior performance of NSXCF in this environment suggests that the model is well-suited for scenarios where interaction data is rich, even if the dataset is not large in terms of the total number of users or items.

The higher Normalized Discounted Cumulative Gain (NDCG) scores of NSXCF in both NDCG@10 and NDCG@20 suggest that the model effectively prioritizes the most relevant items for users. This is particularly important in scenarios such as movie recommendations, where users often expect the system to accurately identify their top interests. Compared to other models like RLL and XGBoost, NSXCF shows a noticeable increase in NDCG metrics, which emphasizes its capacity to generate highly personalized recommendations by leveraging both symbolic attributes and collaborative filtering signals.

For the ML-100k dataset, NSXCF achieves an NDCG@10 of 0.5233, outperforming the other models. This is significant because dense datasets like ML-100k allow the model to exploit the abundance of user-item interactions to provide more precise rankings. The superior ranking performance for the ML-100k dataset highlights NSXCF's robustness in environments rich with user behavior data, making it a promising model for platforms that have access to extensive historical user interactions.

In terms of accuracy, the NSXCF model achieved the highest Area Under the Curve (AUC) score for the Taobao dataset and the third-highest for the ML-100k dataset. The Taobao dataset, being less dense with a larger number of users and items, poses different challenges, such as the cold-start problem and sparsity. Despite these challenges, the NSXCF model performs exceptionally well, achieving an AUC of 0.67, significantly higher than the competing models like XGBoost (0.52). This implies that NSXCF is better equipped to handle sparsity by leveraging symbolic rules and collaborative signals, which makes it versatile for real-world applications where data density varies significantly across users and items.

The comparison between the two datasets also reveals interesting patterns. In the ML-100k dataset, NSXCF slightly underperforms the NSICF in AUC (0.73 vs. 0.80), indicating that in highly dense scenarios, attribute-only approaches may sometimes be advantageous for binary relevance tasks. However, NSXCF's competitive performance across multiple metrics suggests that the integration of both attribute reasoning and collaborative filtering offers balanced strengths, providing superior recommendations in terms of ranking quality while also maintaining competitive accuracy in predicting user preferences.

The following table 5.2 summarizes the performance of all models across the two datasets:

Table 5.2: Metrics of Recommendation Scores Across Compared Models and the Datasets of ML-100k and Taobao

Dataset	**Metrics**	**RRL**	**XGBoost**	**NSICF**	**NSXCF**
ML-100K	NDCG@10	0.412	0.4202	0.4171	**0.5233**
	NDCG@20	0.581	0.537	0.645	**0.665**
	AUC	0.74	0.69	**0.80**	0.73
Taobao	NDCG@10	0.298	**0.371**	0.311	0.365
	NDCG@20	0.362	**0.411**	0.342	0.369
	AUC	0.55	0.52	0.61	**0.67**

These findings indicate that NSXCF is highly competitive in terms of ranking performance and accuracy, with particular strengths in handling sparse datasets through its neuro-symbolic hybrid approach.

5.6 Ablation Study

The ablation study aims to evaluate the contribution of different components of the NSXCF model. By systematically removing specific parts of the model, we aim to observe how each influences the final recommendation performance. This approach helps to determine the value added by each individual component, providing deeper insights into the architecture's efficiency.

Initially, we removed the attribute rule generation component, relying solely on collaborative filtering for the recommendation task. The results, as shown in Table 5.3, indicate a significant decrease in performance, particularly in terms of ranking metrics such as NDCG@10 and NDCG@20. Specifically, the NDCG@10 score dropped from 0.5233 to 0.410 for the ML-100k dataset, and from 0.365 to 0.154 for the Taobao dataset. This significant

drop illustrates the importance of incorporating attributes, which provide valuable context for generating personalized and relevant recommendations.

Next, we examined the impact of excluding the graph neural network (GNN) layer by using only the linear layer for reading user and item IDs. The GNN plays a crucial role in capturing higher-order connections in the data, which are essential for understanding complex relationships between users and items. When the GNN layer was removed, we observed further declines in model performance. The NDCG@20 for the ML-100k dataset dropped to 0.247, compared to 0.665 when the GNN was included, highlighting how critical the GCN architecture is for enhancing the model's ability to understand user preferences.

Furthermore, the absence of the embedding layer led to a significant reduction in the AUC score for the Taobao dataset, from 0.67 to 0.521, which indicates that the embedding layer is crucial for learning effective feature representations, especially for large, sparse datasets. These findings confirm that both the inclusion of attribute rule generation and the GNN layer are essential for achieving optimal performance with the NSXCF model.

Table 5.3: Ablation Study of the Model Components

Methods	ML-100k			Taobao		
	NDCG@10	NDCG@20	AUC	NDCG@10	NDCG@20	AUC
Without Attributes	0.410	0.497	0.618	0.154	0.235	0.586
Without Embedding	0.205	0.247	0.587	0.110	0.148	0.521

The ablation study demonstrates the necessity of each component within the NSXCF model. The inclusion of attribute rules, the GNN layer, and the embedding layer collectively contribute to the model's strength in delivering personalized and accurate recommendations, which is reflected in its superior performance compared to state-of-the-art alternatives.

5.7 Attribute Rule Explanation Style Analysis

In this section, we analyze how the generated attribute conjunction rules align with the recommendations made by the NSXCF model. The evaluation considers different thresholds for the recommendation scores—0.6, 0.7, and 0.8—as hyperparameters. By systematically varying these thresholds, we assess the granularity of the recommendations and how well the item attributes in the generated rules correspond to the recommended items. Additionally, we display the characteristics of the data to highlight the diverse contexts in which these rules operate.

The NSXCF model's logical layer was configured to produce 100 rules, from which redundant rules were eliminated, yielding 52 unique rules for the ML-100k dataset and 57 for the Taobao dataset. To further dissect these rules, we calculated the average number of attributes present for both users and items. This analysis provides insights into the complexity of the generated rules and the interplay between user preferences and item characteristics. We subsequently measured the degree of matching between these attributes and the recommended items.

The characteristics of the datasets play a pivotal role in understanding the results. The ML-100k dataset consists of 100,000 ratings from 943 users on 1,682 items, resulting in a

high density of interactions. This density allows for richer user profiles and more diverse item attributes to be incorporated into the rules. In contrast, the Taobao dataset is more extensive, with a larger number of items (over 1 million) and users (approximately 500,000), leading to sparsity in user-item interactions. This disparity necessitates the model to leverage different strategies for attribute generation and matching.

As shown in Table 5.4, the results are consistent across both datasets and exhibit a notable improvement as the recommendation score threshold is lowered. A lower threshold enables the model to generate more inclusive rules, thereby increasing the average score of matching attributes.

Table 5.4: Attribute Rules Matching to the Final Recommendation Attributes

Metrics	ML-100k			Taobao		
	0.6	0.7	0.8	0.6	0.7	0.8
Number of Rules	52			57		
Avg. Number of User Attributes	14			2		
Avg. Number of Item Attributes	8			21		
Avg. Score of Matching	0.84	0.61	0.49	0.79	0.75	0.52

Attributes Rule Sample Interpretation. To provide a clearer understanding of how user attributes influence recommendations, we present a sample of six rules from the 52 extracted conjunction rules of the ML-100k dataset. For example, Rule 1 suggests that if a user is female (Gender_F) and works in healthcare (Occupation_Doctor/Health Care), then romance movies are the most likely recommendation. This example illustrates how the model leverages user attributes to generate personalized recommendations tailored to individual preferences.

- Rule 1: Gender_F $\wedge$ Occupation_Doctor/Health Care $\wedge$ *Romance*
- Rule 2: *Comedy* $\wedge$ *Fantasy* $\wedge$ *Film-Noir* $\wedge$ Occupation_Homemaker
- Rule 3: *Adventure* $\wedge$ *Comedy* $\wedge$ Occupation_College/Grad Student $\wedge$ *Thriller*
- Rule 4: *Adventure* $\wedge$ *Comedy* $\wedge$ *Drama* $\wedge$ *Fantasy* $\wedge$ Occupation_Artist
- Rule 5: *Comedy* $\wedge$ *Fantasy* $\wedge$ Occupation_Customer Service
- Rule 6: *Animation* $\wedge$ *Comedy* $\wedge$ Occupation_Other $\wedge$ *War*

Furthermore, we examined the implications of varying thresholds on the rule generation process. Lowering the threshold results in more generalized recommendations that may capture a broader audience, while higher thresholds yield specific rules that cater to niche segments. For example, while the average score of matching attributes is high at a threshold of 0.6, it decreases as the threshold is raised to 0.8. This phenomenon suggests that while specificity can enhance precision, it may also reduce the overall applicability of the rules across a diverse user base.

To quantify the effectiveness of these attribute rules, we employed metrics such as precision and recall in addition to matching scores. These metrics allow us to assess how well the generated rules align with user preferences over time, leading to a better understanding of how the NSXCF model can be utilized in practical recommendation scenarios.

In conclusion, the analysis of attribute rule explanations reveals the efficacy of the NSXCF model in producing understandable and relevant recommendations. By grounding these recommendations in user attributes and leveraging varying thresholds, we can ensure that the recommendations remain both personalized and interpretable, enhancing user experience and trust in the system.

5.8 Similar Item Explanation Style Analysis

This section evaluates the effectiveness of similar item recommendations by measuring how closely the recommended items align with the user's historical preferences. We specifically focus on the top 20 similar items and assess the proportion of these items that were part of the user's past choices. The results, presented in Table 5.5, indicate that increasing the number of top similar items generally enhances the match rate, particularly for the Taobao dataset, where the match score reaches 0.8.

The analysis focuses on two datasets: ML-100k, which contains a rich set of user-item interactions, and Taobao, known for its larger number of items and users. By examining the top 10 and top 20 similar items, we can gauge how well the model captures user preferences across varying numbers of recommendations.

As demonstrated in Table 5.5, the match rates improve significantly as we increase the number of recommended items. Specifically, the average match rate for the top 10 similar items is 0.2 for the ML-100k dataset and 0.4 for the Taobao dataset. In contrast, the match rates for the top 20 similar items rise to 0.6 for ML-100k and 0.8 for Taobao, indicating a strong correlation between the number of similar items recommended and the likelihood that these items align with users' historical choices .

Table 5.5: Average Number of Similar Items

Similarity	ML-100k	Taobao
Top 10	0.2	0.4
Top 20	0.6	0.8

These findings underscore the importance of providing a sufficient number of recommendations to enhance user satisfaction and engagement. The model's ability to recommend items that users have previously interacted with reinforces its effectiveness and accuracy. Overall, the results highlight the potential for utilizing similar item recommendations to improve user experience and drive higher engagement levels in both datasets.

5.9 Conclusion

In this chapter, we have presented a detailed analysis of the NSXCF model's performance across several metrics and datasets. Our model demonstrates strong performance in dense datasets, like ML-100k, and maintains high accuracy in less dense datasets, such as Taobao. The ablation study highlights the critical importance of both the attribute data and the GNN layer in achieving optimal recommendations. The attribute rule analysis and similar item explanation analysis provide further insights into the interpretability and effectiveness of our model. These findings contribute to the broader understanding of how hybrid models, which combine collaborative filtering with additional data, can enhance recommendation quality. Future work could explore the scalability of this approach to even larger and more complex datasets.

General Conclusion

Summary of Findings

This thesis introduced the Neuro-Symbolic Collaborative Filtering NSXCF model, designed to bridge the gap between accuracy and explainability in recommendation systems. By integrating neural networks with symbolic reasoning, the NSXCF model offers not only precise recommendations but also interpretable explanations through attribute rules and similar item suggestions. The experimental evaluation, conducted on the Taobao and MovieLens (ML-100K) datasets, demonstrated the model's superior performance in diverse recommendation environments. The NSXCF model consistently outperformed traditional and state-of-the-art explainable AI models in key metrics such as NDCG, AUC, and rule accuracy. This suggests that combining collaborative filtering with logical reasoning can significantly enhance the quality and interpretability of recommendations, making NSXCF a robust solution for modern recommendation challenges.

Limitations and Challenges

Despite its promising results, the NSXCF model faces certain limitations and challenges. One of the primary challenges lies in the complexity of training, as the integration of symbolic reasoning with neural networks demands significant computational resources. Additionally, the binarization of attributes, while effective for rule generation, might lead to information loss, particularly in scenarios with nuanced or continuous attribute values. Another limitation is the model's dependency on well-defined attribute sets; in cases where user or item attributes are sparse or noisy, the performance of the NSXCF model could be adversely affected. Finally, while the model provides explanations through attribute rules and similar items, these explanations might not always align with user expectations, especially in highly personalized recommendation scenarios.

Future Directions for Research

To build on the findings of this research and address the identified limitations, several future research directions are proposed:

- **Enhance the Hyper-parameter Optimization Approach:** Future work could focus on developing more sophisticated hyper-parameter tuning methods, potentially

leveraging techniques like Bayesian optimization or automated machine learning (AutoML) to improve model performance across different datasets.

- **Explore More Advanced Feature Selection Algorithms:** Investigating more advanced or domain-specific feature selection algorithms could help in better capturing the nuances of user and item attributes, thereby improving both recommendation accuracy and explainability.
- **Incorporate Multi-modal Data:** Expanding the model to incorporate multi-modal data, such as text, images, or user behavior logs, could enhance its ability to make recommendations in more complex and real-world scenarios.
- **Scalability Improvements:** Research efforts could be directed towards improving the scalability of the NSXCF model, enabling it to handle larger datasets and more complex recommendation environments without a proportional increase in computational resources.

In conclusion, the NSXCF model marks a significant step forward in the domain of explainable AI for recommendation systems. By addressing its current limitations and exploring the proposed future research directions, the model's applicability and impact could be further expanded, paving the way for more transparent, accurate, and user-friendly recommendation systems.

Bibliography

[1] Chen Gao et al. "A Survey of Graph Neural Networks for Recommender Systems: Challenges, Methods, and Directions." In: *ACM Trans. Recomm. Syst.* 1.1 (2023). DOI: 10.1145/3568022. URL: https://doi.org/10.1145/3568022.

[2] Yelp Inc. *Yelp Dataset*. https://www.yelp.com/dataset. Accessed: April 17, 2024. 2018.

[3] Khalid al Fararni et al. "Comparative Study on Approaches of Recommendation Systems." In: Apr. 2020, pp. 753–764. ISBN: 978-981-15-0946-9. DOI: 10.1007/978-981-15-0947-6_72.

[4] Michael J. Pazzani and Daniel Billsus. "Content-Based Recommendation Systems." In: *The Adaptive Web: Methods and Strategies of Web Personalization*. Ed. by Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 325–341. ISBN: 978-3-540-72079-9. DOI: 10.1007/978-3-540-72079-9_10. URL: https://doi.org/10.1007/978-3-540-72079-9_10.

[5] Jonathan Herlocker et al. "Evaluating collaborative filtering recommender systems." In: *ACM Transactions on Information Systems (TOIS)* 22.1 (2004), pp. 5–53.

[6] Yehuda Koren, Robert Bell, and Chris Volinsky. "Matrix Factorization Techniques for Recommender Systems." In: *Computer* 42.8 (2009), pp. 30–37. DOI: 10.1109/MC.2009.263.

[7] Robin Burke. "Hybrid Recommender Systems: Survey and Experiments." In: *User Modeling and User-Adapted Interaction* 12.4 (2002), pp. 331–370. ISSN: 1573-1391. DOI: 10.1023/A:1021240730564. URL: https://doi.org/10.1023/A:1021240730564.

[8] Thiago Silveira et al. "How good your recommender system is? A survey on evaluations in recommendation." In: *International Journal of Machine Learning and Cybernetics* 10.5 (2019), pp. 813–831. ISSN: 1868-808X. DOI: 10.1007/s13042-017-0762-9. URL: https://doi.org/10.1007/s13042-017-0762-9.

[9] Yongfeng Zhang and Xu Chen. "Explainable Recommendation: A Survey and New Perspectives." In: *Foundations and Trends® in Information Retrieval* 14.1 (2020), pp. 1–101. ISSN: 1554-0669. DOI: 10.1561/1500000066. URL: http://dx.doi.org/10.1561/1500000066.

[10] Francesco Ricci, Lior Rokach, and Bracha Shapira. "Introduction to Recommender Systems Handbook." In: *Recommender Systems Handbook*. Ed. by Francesco Ricci et al. Boston, MA: Springer US, 2011, pp. 1–35. ISBN: 978-0-387-85820-3. DOI: 10.1007/978-0-387-85820-3_1. URL: https://doi.org/10.1007/978-0-387-85820-3_1.

[11] Sean McNee, John Riedl, and Joseph Konstan. "Being accurate is not enough: how accuracy metrics have hurt recommender systems." In: *CHI'06 extended abstracts on human factors in computing systems*. ACM. New York, 2006, pp. 1097–1101.

[12] Graham Peake and Jia Wang. "Explanation mining: Post hoc interpretability of latent factor models for recommendation systems." In: *Proceedings of Beyond Personalization 2005: A Workshop on the Next Stage of Recommender Systems Research at the 2005 International Conference on Intelligent User Interfaces*. ACM. San Diego, CA, USA, 2018, pp. 2060–2069.

[13] Jie Zhou et al. *Graph Neural Networks: A Review of Methods and Applications*. 2021. arXiv: 1812.08434 [cs.LG]. URL: https://arxiv.org/abs/1812.08434.

[14] Jie Zhou et al. "Graph neural networks: A review of methods and applications." In: *AI Open* 1 (2020), pp. 57–81. ISSN: 2666-6510. DOI: https://doi.org/10.1016/j.aiopen.2021.01.001. URL: https://www.sciencedirect.com/science/article/pii/S2666651021000012.

[15] Steffen Rendle et al. "BPR: Bayesian personalized ranking from implicit feedback." In: *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*. 2009, pp. 452–461.

[16] Xavier Glorot and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Yee Whye Teh and Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, 2010, pp. 249–256. URL: https://proceedings.mlr.press/v9/glorot10a.html.

[17] Rex Ying et al. *GNNExplainer: Generating Explanations for Graph Neural Networks*. 2019. arXiv: 1903.03894 [cs.LG]. URL: https://arxiv.org/abs/1903.03894.

[18] Chris Lin et al. *Graph Neural Networks Including Sparse Interpretability*. 2020. arXiv: 2007.00119 [cs.LG]. URL: https://arxiv.org/abs/2007.00119.

[19] Minh N. Vu and My T. Thai. *PGM-Explainer: Probabilistic Graphical Model Explanations for Graph Neural Networks*. 2020. arXiv: 2010.05788 [cs.LG]. URL: https://arxiv.org/abs/2010.05788.

[20] Dongsheng Luo et al. *Parameterized Explainer for Graph Neural Network*. 2020. arXiv: 2011.04573 [cs.LG]. URL: https://arxiv.org/abs/2011.04573.

[21] Hao Yuan et al. "XGNN: Towards Model-Level Explanations of Graph Neural Networks." In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '20. ACM, Aug. 2020. DOI: 10.1145/3394486.3403085. URL: http://dx.doi.org/10.1145/3394486.3403085.

[22] Xiang Wang et al. "Neural Graph Collaborative Filtering." In: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR'19. Paris, France: Association for Computing Machinery, 2019, pp. 165–174. ISBN: 9781450361729. DOI: 10.1145/3331184.3331267. URL: https://doi.org/10.1145/3331184.3331267.

[23] Xiangnan He et al. "LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation." In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '20. Virtual Event, China: Association for Computing Machinery, 2020, pp. 639–648. ISBN: 9781450380164. DOI: 10.1145/3397271.3401063. URL: https://doi.org/10.1145/3397271.3401063.

[24] Petar Veličković et al. *Graph Attention Networks*. 2018. arXiv: 1710.10903 [stat.ML].

[25] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762.

[26] Xiao Wang et al. *Multi-Component Graph Convolutional Collaborative Filtering*. 2019. arXiv: 1911.10699 [cs.LG]. URL: https://arxiv.org/abs/1911.10699.

[27] William L. Hamilton, Rex Ying, and Jure Leskovec. *Inductive Representation Learning on Large Graphs*. 2018. arXiv: 1706.02216 [cs.SI]. URL: https://arxiv.org/abs/1706.02216.

[28] Rex Ying et al. "Graph Convolutional Neural Networks for Web-Scale Recommender Systems." In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '18. ACM, July 2018. DOI: 10.1145/3219819.3219890. URL: http://dx.doi.org/10.1145/3219819.3219890.

[29] Zichao Wang et al. "Transparent Classification with Multilayer Logical Perceptrons and Random Binarization." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04. 2020, pp. 6331–6339. DOI: 10.1609/aaai.v34i04.6102.

[30] Zhuo Wang et al. *Scalable Rule-Based Representation Learning for Interpretable Classification*. 2021. arXiv: 2109.15103 [cs.LG]. URL: https://arxiv.org/abs/2109.15103.

[31] Tianchi. *Ad Display/Click Data on Taobao.com*. Accessed: 2024-08-10. 2018. URL: https://tianchi.aliyun.com/dataset/dataDetail?dataId=56.

[32] F. M. Harper and J. A. Konstan. "The MovieLens Datasets: History and Context." In: *ACM Transactions on Interactive Intelligent Systems (TiiS)* 5.4 (2015), 19:1–19:19. DOI: 10.1145/2827872.

[33] Olivier Jeunen, Igor Potapov, and Andrey Ustimenko. *On (Normalised) Discounted Cumulative Gain as an Off-Policy Evaluation Metric for Top-n Recommendation*. arXiv preprint arXiv:2307.15053. 2024. arXiv: 2307.15053 [cs.IR]. URL: https://arxiv.org/abs/2307.15053.