

# **DESIGN AND DEVELOPMENT OF RECOMMENDATION SYSTEM USING GRAPH NEURAL NETWORK**

MTech Thesis

Submitted In Partial Fulfilment of The Requirements  
For The Award of The Degree  
of

**MASTER OF TECHNOLOGY**

in

**SOFTWARE ENGINEERING**

submitted by:

**VIJAY KUMAR**  
**(2K23/SWE/12)**

Under the guidance of

**Dr. Ruchika Malhotra**

**Head of Department**

**Department of Software Engineering**



**DEPARTMENT OF SOFTWARE ENGINEERING**

**DELHI TECHNOLOGICAL UNIVERSITY**

(Formerly Delhi College of Engineering)

Bawana Road, Delhi - 110042

JUNE, 2025

**DEPARTMENT OF SOFTWARE ENGINEERING**  
**DELHI TECHNOLOGICAL UNIVERSITY**  
(Formerly Delhi College of Engineering)  
Bawana Road, Delhi-110042

**ACKNOWLEDGEMENT**

I am very thankful to **Prof. Ruchika Malhotra** (Head of Department, Department of Software Engineering) and all the faculty members of the Department of Software Engineering at DTU. For the project, all of them provided me with plenty of guidance and encouragement. I also want to thank the University for giving me the environment, testing facilities, infrastructure, and labs that made it possible for me to work without any problems. Additionally, I would like to thank our peer group, seniors, and lab assistants for their assistance, lending me their extensive knowledge on a variety of subjects.

Place: Delhi  
Date:

VIJAY KUMAR  
(2K23/SWE/12)

**DEPARTMENT OF SOFTWARE ENGINEERING**  
**DELHI TECHNOLOGICAL UNIVERSITY**  
(Formerly Delhi College of Engineering)  
Bawana Road, Delhi - 110042

**CANDIDATE'S DECLARATION**

I, Vijay Kumar, Roll No. 2K23/SWE/12 student of M. Tech (Software Engineering), hereby declare that the Project Dissertation title is " **Design and Development of Recommendation System Using Graph Neural Network**". It is being presented to Delhi Technological University as a legitimate report of my MTech thesis, partially fulfilling the requirements for the award of the M.Tech (Software Engineering) degree. The thesis's content hadn't been submitted to any university or other organisation for consideration for a degree.

Place: Delhi  
Date:

Vijay Kumar  
(2K23/SWE/12)

This is to certify that the student has incorporated all the corrections suggested by the examiner in the thesis and the statement made by the candidate is correct to the best of our knowledge.

**Signature of Supervisor**

**Signature of External Examiner**

**DEPARTMENT OF SOFTWARE ENGINEERING**  
**DELHI TECHNOLOGICAL UNIVERSITY**  
(Formerly Delhi College of Engineering)  
Bawana Road, Delhi-110042

**CERTIFICATE**

I, hereby certify that the Project titled " **Design and Development of Recommendation System Using Graph Neural Network** " which documents the work that Vijay Kumar, who is Roll No. 23/SWE/12, Department of Software Engineering at Delhi Technological University, Delhi, completed under my supervision in order to partially fulfil the requirements for being awarded of the degree Master of Technology (Software Engineering). To the finest of my knowledge, neither this university nor any other has ever accepted the work in whole or in part for a certificate or degree.

Place: Delhi

Date:

Dr. Ruchika Malhotra  
Head of Department  
Department of Software Engineering  
Delhi Technological University

## **ABSTRACT**

As current communication technology has advanced, e-commerce has become increasingly prevalent. These days, people can purchase goods from the comfort of their homes through online retailers like Amazon, Flipkart, and others, as well as find entertainment on websites like YouTube and Spotify. Another issue of "spoilt for choices" arises because internet platforms may accommodate a vast number of things from which users can choose. Recommender systems have become a very powerful tool to address this problem; given the user's past behaviour and the attributes of the various items, these systems apply an algorithm to create a candidate set from the entire set of items, rank the items in the candidate sets, and then display the items to the user based on the item ranking.

We provide a thorough analysis of a recommender system in this dissertation. We go over the model that underpins it, its phases, the algorithms that drive these systems—whether they are sophisticated techniques based on deep learning and machine learning

or more conventional approaches like matrix factorization—as well as the advantages and disadvantages of each. We also go over the applications of recommender systems. As the internet's reach increases and more people begin consuming digital material and engaging in e-commerce, we have concentrated on session-based recommender systems. Both consumers and producers benefit from a session-based recommender system since it improves the buying experience for consumers and allows businesses to make better decisions by analysing the data produced by the system and using that information to improve the recommender systems' performance, which in turn improves the customer experience. We're putting forth a brand-new approach for session-based recommender systems. For our model, graphical neural networks (GNN) are being used. After preprocessing the user-item interaction datasets, our model learns item and positional embeddings, learns each item's relevant neighbourhood using item-KNN, generates a local and global graph, and obtains an embedding of each item in both local and global contexts. It then adds the local and global embeddings of the item to obtain the final representation, and uses the dot product of the final representation and the initial item embedding to obtain a final score that indicates the item's significance to the user. Lastly, it suggests items to the user depending on the final score.

We have assessed our model's efficacy using precision ( $P@K$ ) and mean reciprocal rank ( $MRR@K$ ). Diginetica, TMall, and Nowplaying datasets were used. Our approach outperforms all non-graphical neural network-based recommendation systems. Compared to other GNN-based models, ours performs significantly better.

# Content

<b>CANDIDATE DECLARATION</b>	<b>I</b>
<b>CERTEFICATE</b>	<b>II</b>
<b>ABSTRACT</b>	<b>III</b>
<b>CONTENT</b>	<b>V</b>
<b>ACKNOWLEDGEMENT</b>	<b>IV</b>
<b>Abbreviations List .....</b>	<b>IX</b>
<b>1 INTRODUCTION</b>	<b>1</b>
1.1 Recommendation System Phases .....	2
1.1.1 Phase-I Information Collection .....	3
1.1.2 Phase-II Learn phase.....	4
1.1.3 Phase-III Predict phase .....	4
1.2 Objective of this Dissertation.....	4
1.3 Structure of this Dissertation .....	4
<b>2 LITERATURE REVIEW</b>	<b>5</b>
2.1 Recommender system technique .....	5
2.1.1 Content-based recommendation System.....	5
2.1.2 Collaborative-Filtering-based recommender System.....	6
2.1.3 Hybrid-filtering-based recommendation system.....	8
2.2 Deep Learning based Recommender Systems.....	8
2.3 Challenges in Deep Learning based Recommender Systems .....	9
2.4 Session-based Recommendation System.....	10
2.5 Components of an SBRS .....	11
2.5.1 User.....	11
2.5.2 Items .....	11
2.5.3 Action .....	11
2.5.4 Session.....	11
2.6 Related Work .....	13
2.7 Applications.....	13
2.8 Challenges faced in SBRS .....	14
2.9 Graphical Neural Networks .....	14
<b>3 METHODOLOGY</b>	<b>17</b>
3.1 Algorithm for our proposed model .....	17
<b>4 RESULT AND EXPERIMENTAL ANALYSES</b>	<b>23</b>
4.1 Datasets used .....	23

4.2	Data Preprocessing .....	23
4.3	Evaluation metrics .....	24
4.4	Parameter setup .....	24
4.5	Item-KNN Parameter.....	24
4.6	Baseline Algorithms .....	25
4.7	Result.....	25
4.8	Observations .....	28
<b>CONCLUSION</b>		<b>29</b>
<b>REFERENCES</b>		<b>30</b>

# List of Figures

1.1	User-centric vs Item-centric Recommender Systems . . . . .	1
1.2	Recommender System Phases. . . . .	3
2.1	Content-based Filter . . . . .	5
2.2	Collaborative Filter . . . . .	6
2.3	Session-Based Recommender System. . . . .	11
2.4	Graphical Neural Network. . . . .	15
3.1	Flow chart of the proposed model. . . . .	18
3.2	Global Graph . . . . .	19
3.3	Local graph . . . . .	20
4.1	Impact of Dropout ratio . . . . .	27
4.2	Stabilization of loss function with epoch for TMall dataset. . . . .	27
4.3	Trajectory of P@20 vs epochs for TMall dataset . . . . .	27



# List of Tables

I	User-Item Matrix . . . . .	2
II	Benefits of Recommender Systems . . . . .	2
III	Machine Learning Techniques for Recommender System . . . . .	8
III	A reviewed publications lookup table. . . . .	9
IV	Deep Learning based models with applications. . . . .	9
V	Difference between session data & sequence data . . . . .	10
VI	SBRS sub-areas . . . . .	12
VII	Techniques for Session-based Recommender Systems . . . . .	13
VIII	Dataset statistics after preprocessing . . . . .	23
IX	Baseline algorithms . . . . .	25
X	Results for @20 . . . . .	26
XI	Results for @10 . . . . .	26
XII	Performance of our model when key components were removed on a sin- gle basis . . . . .	26
XIII	Effects of different aggregation operations. . . . .	27

## List of Abbreviations

- NLP – Natural Language Processing
- GNN – Graphical Neural Network
- GRU – Gated Recurrent Unit
- ML – Machine Learning
- SBRS – Session-Based Recommendation System
- RNN – Recurrent Neural Network
- RS – Recommendation System

# CHAPTER 1

## INTRODUCTION

A recommender system is an algorithmic method on the computer that generates personalized recommendations through the analysis of user behavior and item features. A recommender system utilizes various methods, such as collaborative filter, content-based filter, and hybrid models, in order to provide recommendations to users. Recommender systems make precise predictions regarding user preference by measuring similarity, past behavior, and users' opinions. These are necessary to drive consumer interaction, enhance user experience, and enhance business revenue.

They are extensively used in online shopping, social networks, streaming websites, and everywhere else where customized recommendations can be used to improvise user experience and boost activity. As algorithms for machine learning and data mining get more advanced, recommender systems evolve continuously to provide users with ever-better and relevant recommendations.

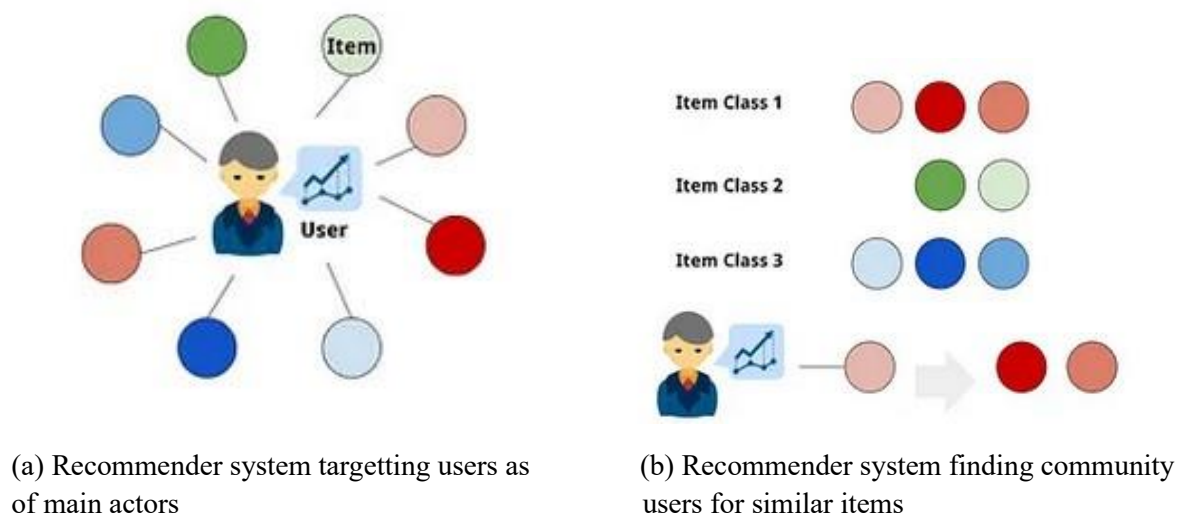


Figure 1.1: User-centric vs Item-centric Recommender Systems

The main two actors in a recommender system are-

- Users: user is the actor "for whom" the recommendation is made. For eg., users of social media websites like TikTok.
- Items: item is the actor "of which" the recommendation is made. For eg., on platforms like youtube, videos are recommended to users.

A recommender system is built on top of the user-item matrix, which shows the relationship between users and items. A user-item matrix is shown in Table I, which contains five users and five items. Using a scale of 1 to 5, users rate items.

The likelihood that a user has enjoyed the item increases with the rating. As we can see, only 12 of the 25 cells in total have a rating value. This indicates that not every user rated every item; some users only rated objects with which they interacted in some way.

Table I: User-Item Matrix

User/Item	First Item	Second Item	Third Item	Fourth Item	Fifth Item
First User		2		1	
Second User	5			1	4
Third User	2	4			
Forth User				2	3
Fifth User		2		4	3

Predicting a value for these empty cells and then suggesting products that the user has rated higher is the fundamental function of a recommender system. For instance, if User 4 has only rated Items 4 and 5, the recommender system's job is to anticipate User 4's rating for Items 1, 2, and 3, and then suggest to User 4 the item for which the expected rating is greater. By utilising similarity metrics and user-item interactions, these recommendations are produced, increasing user engagement and happiness.

Table II: Benefits of Recommender Systems

Benefits for Users	Benefits for Businesses
<ul style="list-style-type: none"> <li>• Personalized Recommendations</li> <li>• Time-saving</li> <li>• Discovery of New Items</li> <li>• Enhanced User Experience</li> <li>• Personalized Notifications</li> </ul>	<ul style="list-style-type: none"> <li>• Increased Sales and Revenue</li> <li>• Improved Customer Satisfaction</li> <li>• Enhanced Customer Engagement</li> <li>• Competitive Advantage</li> <li>• Data-Driven Insights</li> </ul>

## 1.1 Recommendation System Phases

As illustrated in Fig. 1.2, a complete recommendation system is composed of numerous steps that can be broadly divided into three phases. As we can see, a recommendation system

functions as a control system as the feedback mechanism's output is crucial to making additional recommendations.

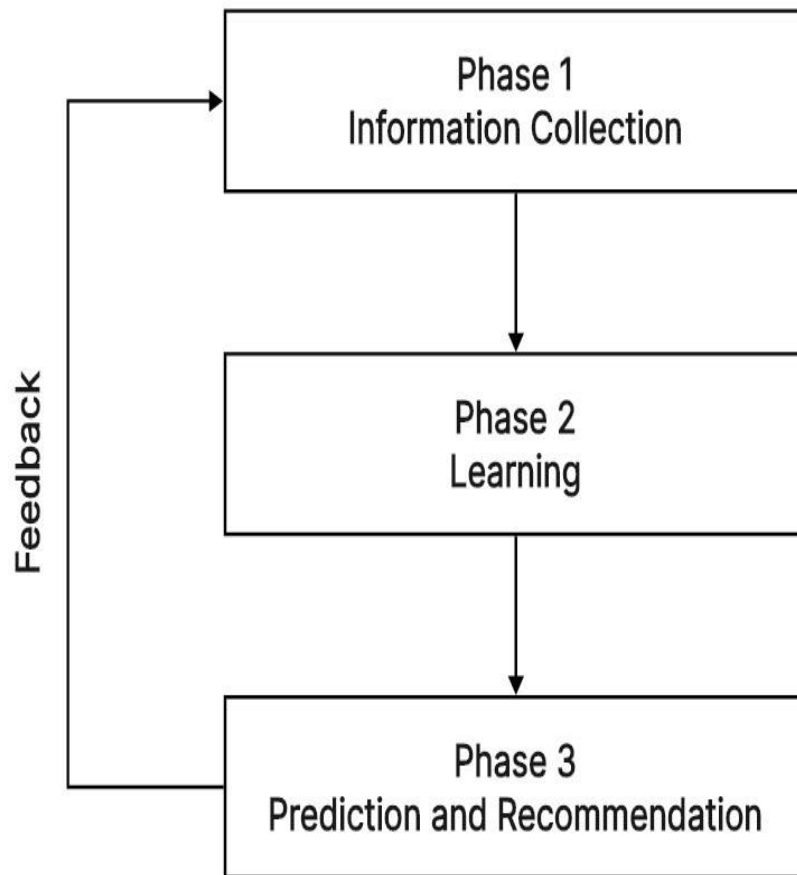


Figure 1.2: Recommender System Phases

### 1.1.1 Phase-I Information Collection

In order to create a user profile, the first step is to gather accurate and pertinent information about the user, including their characteristics, behaviours, and details about the items they access. This stage is crucial for recommendation systems since the created dataset dictates how accurate the recommendations are.

- In order to receive explicit feedback, users must rate products. The data collected is quite trustworthy. Its drawback is that it necessitates additional effort from the user in order to rate.
- Inferring a user's preferences from their behaviour is known as implicit feedback. This is accomplished by tracking user behaviour, including amounts of time spent on websites, links clicked, buttons clicked, and email content, among other things. Implicit feedback has the advantage of requiring no effort from the user, but it may not be as accurate as explicit feedback because it makes assumptions about the user's preferences based on his behaviour. Implicit feedback is seen to be far more objective than explicit input since it does not require the user to behave in a way that is socially acceptable, which helps to lessen prejudice.

- The benefits of both explicit and implicit feedback techniques are combined in hybrid feedback. These two are being combined to create a number of creative strategies. Using implicit rating as a check on explicit rating is one method that can improve accuracy.

### 1.1.2 Phase-II Learn phase

During this stage, the recommendation system uses algorithms to train model and filter the dataset of user profiles. The model learns the user preferences by analysing the user's prior interactions with various items, classifying users with similar users, and classifying products with comparable attributes. Various user and item communities emerge during this stage.

### 1.1.3 Phase- III Predict phase

Using the learned model during the learning phase, our recommendation system provides recommendations in this phase.

We can now ask our recommender system if the user will enjoy an item or not. The model then gives us the probability of the user enjoying the item, and we can apply a threshold to decide if we should recommend it to the user or not.

## 1.2 Objective of this Dissertation

This dissertation intends to explore the fascinating area of recommender systems. Expertise in session-based recommender systems, we intend to conduct an in-depth analysis of recommender systems. We intend to propose a new approach to session-based recommender systems via item-based KNN with graphical neural networks.

## 1.3 Structure of this Dissertation

The structure of this dissertation is as follows:

A recommender system is reviewed in Chapter 2. It presents an in-depth description of the algorithms used in recommender system such as hybrid processes, collaborative filter, and content-based filter. Their limitations, strengths, and applications are explained. Deep learning-based approaches have also been elaborated upon. Session-based recommender systems have also been explained. Definition of an SBRs, its elements, sequence and session data comparison, properties of a session (like time, order, etc.) and their impacts, associated studies, applications, and challenges of SBRs have all been rigorously studied.

Our suggested approach for a session-based recommendation system is presented in Chapter 3. We have provided a detailed process for our algorithm. We go into great detail on the two primary parts of our global graph and session graph. Every equation that was employed is also noted. There is also a flowchart of our suggested model.

Our experiment's specifications and findings are presented in Chapter 4. This chapter presents an analysis of our model's performance in relation to other well-known models. We also present several ablation studies on our model

# CHAPTER 2

## LITERATURE REVIEW

### 2.1 Recommender system technique

Recommender systems can be created using a variety of methods, each with unique advantages and disadvantages.

#### 2.1.1 Content-based Recommendation Systems

By collecting attributes from the contents of items that the user has previously interacted with, content-based recommendation systems leverage user profiles. Consequently, the recommendation metric is similarity.

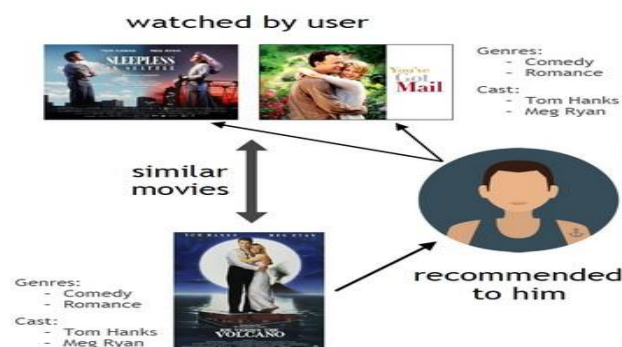


Figure 2.1: Content-based Filtering

The Naïve Bayes classifier, decision trees, cosine-similarity, and other methods can be used to compute similarity. Since recommendation are solely based on the end user's profile, these systems are individualistic in nature. Since only the end-user's profile is used for recommendations and not the profiles of other users who are similar, these systems are serial in nature.

- One advantage of these systems is that they may suggest new products even in the absence of a prior user rating. Additionally, it quickly adapts its recommendations to the user's changing tastes.
- These systems have limited content analysis problems because they rely heavily on item metadata, which should have high-quality descriptions; if item metadata is not rich enough, recommendations suffer. They also face content overspecialisation, where users are presented recommendation that are strikingly similar to item that are already in their profile.
- Examples of such systems include Citeseer, LIBRA.

### 2.1.2 Collaborative-Filtering-based Recommender System

The core algorithm of collaborative filtering is a measure that computes similarity between users and objects at the same time in order to provide recommendations to the user. Since other user options are also utilised, it is parallel in nature. It doesn't depend on the domain. Building a user-item matrix with each row representing a user rating for the item is the fundamental concept underlying this technique. A community is made up of users who are similar to one another.

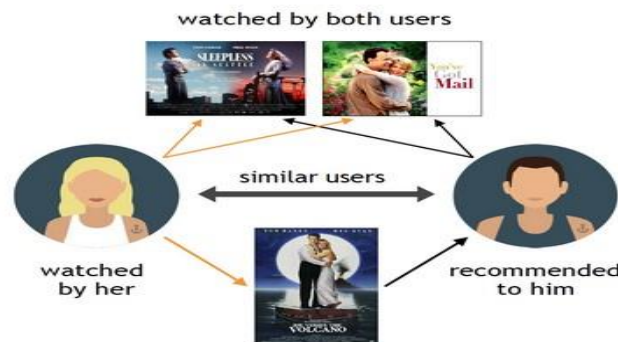


Figure 2.2: Collaborative Filtering

The user receives recommendations for things that other community members have rated but they haven't. In a similar vein, objects that have similar characteristics might be grouped together.

- The entire dataset of user and item interactions is used by memory-based filtering algorithms to produce suggestions. They employ a number of similarity metrics, such as Pearson correlation coefficient, cosine-similarity, etc. Systems that rely on memory are further separated into:
  - User-based filtering: In user-based collaborative filtering, users that share similar traits are grouped together. This is accomplished by computing a metric that represents a user's similarity score. The system will initially find other users who have a high similarity score with the present user before recommending things based on what other similar users are preferring. This kind of recommendation is useful when there is a large user base, such on social media platforms.
  - Filtering based on items Items with comparable features are grouped together in the item-based collaborative filtering technique. If the user has previously rated anything from the group, then other items from the group can also be suggested. When there are a lot of things present, like in e-commerce businesses, this type of strategy is highly helpful. After grouping objects, it suggests groups according to the circumstances.
- Model-based filtering algorithms employ the dataset to create a new model, which is then used to provide more recommendations. Among the methods are:
  - Clustering: In order to find significant communities within datasets, clustering algorithms divide them into groupings [36]. By examining the similarities at two levels—intra-cluster similarity, which should be as high as feasible, and inter-cluster similarity, which should be as low as possible—one may assess the



effectiveness of a clustering technique. By lowering the number of candidates in the recommendation systems, clustering aids.

- Tree graphs, the building blocks of decision trees, are created by replicating a set of training cases for which class label are known and then applying them to previous unseen objects. Decision trees can be used to address some deficient traits.
  - Link analyses: By examining the relationships between related things, a pattern is produced. Web searches make extensive use of it.
  - Regression: One of the most used methods for modelling linear systems is regression. When we wish to forecast the value of a dependent variable whose value is influenced by several independent variables, we utilise this method. The most noticeable aspect of regression is curve fitting.
  - One data mining technique that finds patterns and correlations in a dataset is the association rule.
  - Bayesian classifiers are probabilistic frameworks that use the Bayes theorem and conditional probability as their primary guiding principles for classification [22]. Their ability to handle missing values and withstand isolated noise spots is one of their advantages.
- Collaborative filtering-based recommenders have the advantage of being able to function in domains with low feature density. They can offer haphazard suggestions, like suggesting a product to user M based on comparable user N's likes.
  - Systems based on collaborative filtering have many advantages, but they also present a number of difficulties.
    - Cold start: The "cold-start problem" occurs when a new person or item is added and the recommender system doesn't have enough information to provide precise predictions [10]. Since they lack a user profile, the system is unable to determine the preferences of newly recruited users. Similarly, anytime a new product is released, the system doesn't know enough about its attributes, such if it's popular or how users engage with it. In other words, the matrix becomes sparser and larger as more individuals or objects are added.
    - Scalability: The recommendation system's computational efficiency declines with increasing dataset amount, producing disappointing results [49]. Using methods like Singular Value Reduction (SVD) to lower the dimensionality is one such way.
    - The problem that recommender systems encounter when there are a lot of things present and many of them share similar attributes is called synonymy. It is difficult for a recommendation system to discriminate between things with a lot of identical attributes.
  - Examples of such system includes Ringo, GroupLens, amazon eCommerce store, etc.

### 2.1.3 Hybrid-filtering-based recommendation systems

By merging the two conventional methods of collaborative filtering and content-based filtering, a new area of algorithms has been created. The goal is to employ a different model to counteract the shortcomings of the first one.

- By giving each filtering technique a different weight, weighted hybridisation combines many filtering methods to produce an overall score. P-tango is one instance [13].
- Depending on the circumstance, switching hybridisation employs various strategies. By using a different technique, it overcomes the issue with the first one. DailyLearner is one example.
- Cascade Hybridisation employs a methodical refining process. The following step refines the suggestions made in the previous stage. EntreeC is one such instance [9].
- Mixed hybridisation blends suggestions from various recommendation methods at the item level. PTV is one such instance [65].
- Feature combination is a technique in which the input for the second recommendation system consists of features produced by the first technique. For instance, content-based filtering systems can make use of ratings produced by collaborative filtering. Pipper is one such instance [5].

Table 2.1: Machine Learning Techniques for Recommender Systems

Technique	Research Papers
Collaborative Filtering	[32] [61] [18]
Content-Based Filtering	[50] [46] [21]
Matrix Factorization	[60] [31] [6]
Association Rule Mining	[3] [8] [25]
Factorization Machines	[55] [54]
Probabilistic Graphical Models	[2] [45] [42]

## 2.2 Deep Learning based Recommender Systems

Numerous fields, including image processing [7], healthcare [47], social network analysis [4], and cyber-security in the area of bot detection [63], have been profoundly changed by deep learning. Numerous real-world issues, such as those pertaining to natural language processing, such as the detection of false information and rumours, multilabel text classification, abusive content detection, bot detection, and many more, can be effectively resolved by deep learning-based techniques [35, 33, 34, 64, 58, 44, 59].

Numerous models that make use of neural networks have emerged as a result of recent developments in this field. Neural networks' benefits for recommender systems

- End-to-end Differentiability: The ability of neural architectures to differentiate from one end to the other is a key benefit.
- Leveraging Intrinsic Structure: Deep neural networks are highly skilled at taking advantage of the input data's intrinsic structure.

- Multi-modal and Composite Representation Learning: Deep neural network can handle various neural building block with a single function.

As there are many models based on deep learning, we are providing a lookup table for the various publications of the model in Table II.

Table II: A reviewed publications lookup table.

Categories	Publications
Multilayer Perceptron	[75] [84] [14]
Autoencoder	[19] [38] [39]
Convolutional Neural Networks	[90] [48] [74]
Recurrent Neural Networks	[79] [69]
Neural Attention	[28] [40]
Adversary Network	[70] [72]
Hybrid Models	[12] [20]

The deep learning based models can be applied in various application domains. Table III lists various publications according to their application domains.

Table III: Deep Learning based models with applications

Data Source/Task	Publication
Sequential Information ( User ID w/t)	[11] [79] [80]
Sequential Information (Session based w/o User ID)	[66] [68] [69]
Sequential Information (Check-In, POI)	[74] [84]
Text (Hash Tags)	[23] [48]
Text (Review texts)	[90] [89]
Images (Visual features)	[89] [87]
Audio (Music)	[76] [77]
Video (Videos)	[12] [14]
Social Network	[17] [75]
Cross Domain Network	[17] [75]

## 2.3 Challenges of Deep Learning based Recommendation System

- Interoperability
  - One drawback of deep learning models is their lack of interpretability.
  - Explainability is limited by these networks' non-interpretable hidden weights and activations.
  - Individual neurone interpretation is still difficult.
- Data Requirement
  - It is well known fact that deep learning models requires a lot of data.
  - For deep learning models to support rich parameterisation, there must be enough data.
- Extensive Hyperparameter Tuning
  - Deep learning frequently necessitates extensive hyperparameter tuning.
  - Compared to machine learning models, deep learning adds more hyperparameters.

## 2.4 Session-based Recommendation System

Recommendation systems that base their recommendations on a user's current session or series of actions on a website or application are known as session-based recommendation systems. Session-based recommenders are heavily interested in observing what the user is currently doing, as compared to other recommendation algorithms which tend to operate from a user's entire activity history or profile data for suggestion purposes. They thus provide more timely, relevant, and more responsive to the user's immediate needs and interests recommendations.

Table IV: Difference between session data & sequence data

Data type	Boundary	Order	Time interval	embedded Main relations
Session	Unordered	Multiple	No	Dependencies based on co-occurrence
Session	Ordered	Multiple	Yes	Dependencies based on co-occurrence as well as sequential dependencies also exist
Sequential	Single	Yes	Not included	Sequential dependencies

In order to provide recommendations, session-based recommendation systems frequently look at a user's most recent session data, such as their search queries, page visits, clicks, and other

relevant information. Using this information, the algorithm then suggests items or content that it believes the user would be interested in next.

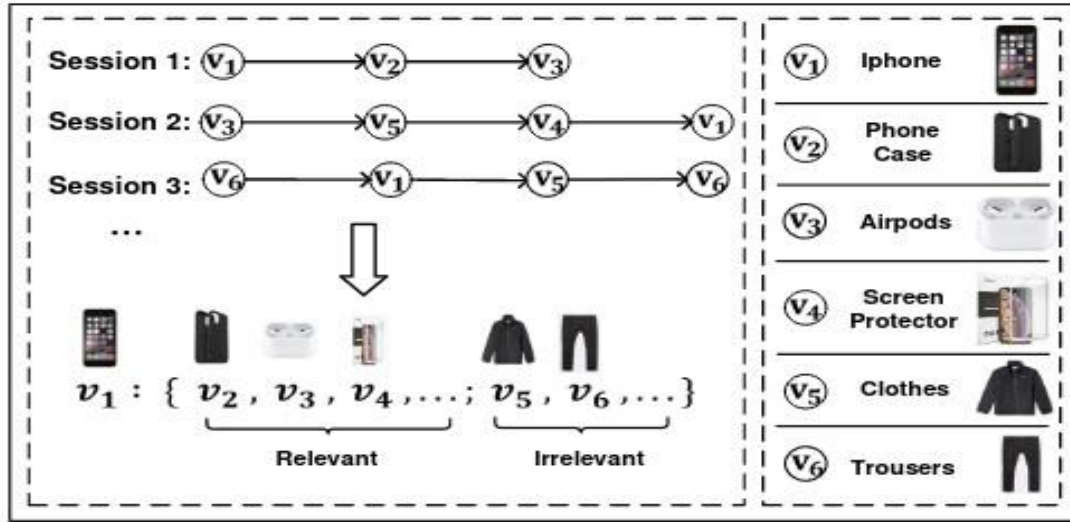


Figure 2.3: Session-Based Recommender System

## 2.5 Components of an SBRS

### 2.5.1 User

A user in a session-based recommendation system is someone who clicks or purchases items, for example, and then gets recommendations as a result. Every user is uniquely identified by their ID and a collection of characteristics. [73]. Some implicit characteristics, such as the user's intentions and moods, may also significantly impact her actions in addition to the obvious qualities. Assume that  $u$  represents a user and that  $U = \{u_1, u_2, \dots, u_n\}$  is the set of all users.

### 2.5.2 Items

Items are the things that should be suggested, such as a song on a music app, a product on an online store, a place to visit, etc. [73]. Assume that  $I = \{i_1, i_2, \dots, i_n\}$  represents an item with all of its attributes grouped together to form a set of objects.

### 2.5.3 Action

An action is when a user click, view, or buy an item. The user interact with the object as a result of the action. Let  $o$  be a tuple expressed as  $o = \langle u, i, a \rangle$ , and let  $a$  represent activities that can be defined based on the issue statement and interaction.

### 2.5.4 Session

In order to create a better recommender system, it is crucial to comprehend the features of the session, which is the most crucial element of a session-based recommendation system.

Table V: SBRS sub-areas

Sub-area	Input	Output	Typical research topic
Next interaction recommendation	The most well-known aspect of this session	Next interaction (item)	Next items, next songs/movies, next POI, next web page, next news, etc.
Next partial session recommendation	The most well-known aspect of this session	Subsequent part of the session	Recommendations for the following things, session/basket completion
Next session recommendation	Historical sessions	Next session	The next package suggestion, the next basket recommendation, etc.

- One of the most crucial characteristics is the length of a session. These fall into three general categories: long, medium, and short [88].
  - More interactions throughout longer sessions yield more contextual input data. However, not every encounter is pertinent, which results in information that is noisy. Incorporating long-term dependence is another challenge.
  - The most prevalent session types on e-commerce platforms are those with a medium duration. They are appropriate in that they often provide the required background information and are less likely to include an excessive number of unrelated encounters.
  - There is very little contextual information in brief sessions. Recommending the session's first interaction is one of the most severe scenarios.
- Order of the session
  - There are several interactions in an ordered session, and there is a strong sequential dependency between them. A user's interaction with an online learning platform, for instance, would be more sequential.
  - Interactions that rely more on one another's co-occurrence make up disordered sessions. Compared to sequential dependencies, these co-occurring dependencies are weak since they are probably confusing. Learning these is really challenging.
- Impact of user actions

- Single-type action sessions consist of only one kind of action, such as clicking or commenting, and hence only one kind of dependency. Modelling such dependencies is simple.
  - Complex dependencies are created by the various types of interactions that occur during several type action sessions. For instance, a user may click, remark, and buy all in the same session.
- Impact of user information
    - Sessions without a user profile are known as anonymous sessions. Only the current session interaction can be used to obtain some contextual information; any prior user interactions are unknown.
    - A session in which a user interacts by revealing their online identify is known as a non-anonymous session. There are their profiles. Their interactions are recorded in their profile as they happen. Gaining insight into the user's long-term preferences is beneficial. You may also examine how their preferences have changed over time. It aids in suggesting better products to them.

## 2.6 Related Work

A lot of work is going on in the field of session-based recommendation systems. Some of which is listed in Table VI.

Table VI: Techniques for Session-based Recommender Systems

Technique	Research Papers
Sequential Models	[26] [91] [29]
Matrix Factorization	[53] [57]
Graph Neural Networks	[81] [24] [41]
Markov Chain	[57] [57]
Convolutional Models	[67] [85] [16]
Self-Supervised Learning	[1] [15] [86]

## 2.7 Applications

SBRs are widely deployed to the benefit of both customers and businesses in a variety of real-world situations and domains.

- Product recommendations at online retailers, such as suggesting a basket of goods or the next item to buy.
- Media and entertainment content recommendations, such as suggesting the next songs to listen to, the next film to watch, the subsequent website to visit, etc.
- In tourism, service recommendations include suggesting the next restaurant to try, the next place of interest, etc.

In addition to these traditional uses, there are several emerging ones, such as suggesting a future investment or trade in the financial sector or a future medical procedure, among others.

## 2.8 Challenges faced in SBRS

SBRSs faces following challenges-

- Sparsity: Because session data is frequently sparse, it can be difficult to precisely record user preferences.
- Cold-start issue: Due to the lack of historical data, it can be challenging to recommend goods to new users or sessions with little data.
- Sequential dependencies: In order to understand user intent and offer pertinent recommendations, recommendations must take into account the order of objects inside a session.
- Data noise: Unrelated interactions may be present in session data, which could lower the calibre of suggestions.
- Scalability: Managing massive session data sets and providing real-time suggestions for a large number of users present scalability issues.
- Contextual information: Recommendations get more complex when time, place, and device are taken into account.
- Evaluation metrics: It can be difficult to choose the right metrics to gauge user satisfaction and take into consideration the recommendations' sequential structure.
- Long-tail items: It can be difficult to suggest less well-liked products with little session data.
- Diversity and serendipity: In session based environments, it can be very difficult to provide fresh and different recommendations to prevent boredom.

## 2.9 Graphical Neural Networks

A type of deep learning model that is specially designed to operate with graph-structural information is called a graphical neural network (GNN), or a GNN. It leverages the interactions and relations that naturally occur in a network to perform different types of prediction, such as node classification, link prediction, and graph-level prediction. The ability of GNNs to extract and utilize structural information in graph data has received a great deal of interest in recent times.

The message transmission mechanism, which enables information flow between connected nodes in a network, lies at the heart of a GNN [78]. By merging data from nearby nodes, the GNN iteratively accumulates and updates node representations. Through this procedure, the model may effectively learn on complicated network structures by capturing both local and global relationships within the graph. The graph convolutional layer is one of the core elements of a GNN [30]. It transforms node representations using a graph-based operation that considers both the node's own properties and those of its neighbours. This enables the GNN to learn significant representations for every node and recognise structural patterns in the graph. Molecular chemistry, computer vision, social network analysis, and recommendation systems are just a few of the fields where GNNs have shown promise. In applications like node classification, where the objective is to forecast the labels or characteristics of certain nodes in a network based on their characteristics and connection patterns, they have shown superior



performance.

In summary, a deep learning model designed specifically for graph-structured data is called a graph neural network (GNN). It captures and makes use of the structural information found in a graph by using graph convolutional layers and message forwarding. GNNs have demonstrated encouraging outcomes in a range of tasks and have found use in a variety of sectors [82].

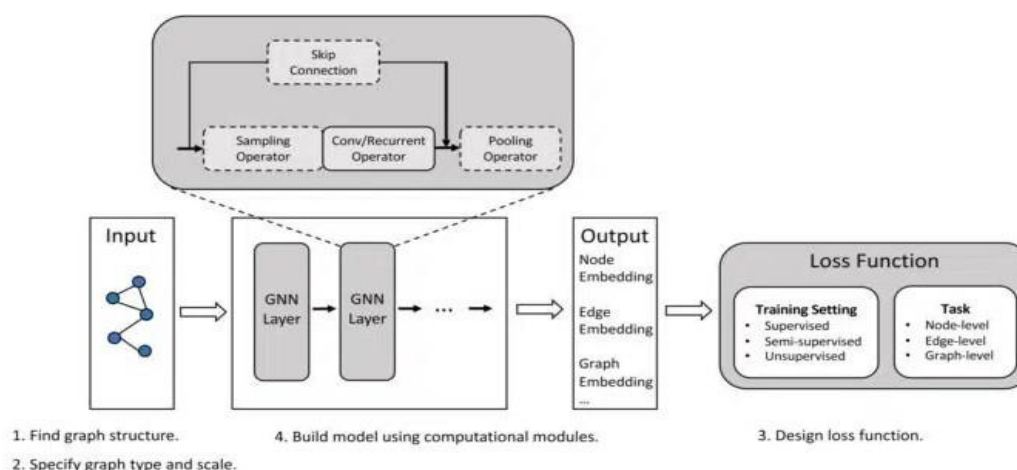


Figure 2.4: Graphical Neural Network

Benefits of graphical neural networks in session-based recommender system-

- **Contextual Information Incorporation:** GNNs are able to automatically take into account contextual information found in session data. GNNs can take advantage of contextual signals like item order, time intervals, and user behaviours by taking into account the relationships and interactions between objects inside a session graph. This enables the recommender system to adjust to changing user preference and offer tailored suggestions according to the situation at hand.
- **Managing Sparsity and Cold-Start:** Session-based recommendation systems face challenges in cold-start scenarios, where there is lack of historical information about a user. GNNs can mitigate this issue by utilising the shared item-item connections across several users' sessions. As GNNs spread information throughout the network, they can impart expertise and make recommendations even to users with little or no history.
- **Improved Recommendation Accuracy:** GNNs can use the collective knowledge from similar sessions to increase recommendation accuracy by identifying both local and global trends in the session graph. GNNs can offer more precise and contextually aware recommendations by combining data from nearby sessions and taking into account their impact on the target session.
- **Scalability and Efficiency:** GNNs are able to effectively scale to large-size session graphs. They can handle massive datasets with millions of sessions and items because they make use of effective message transmission systems and parameter sharing strategies. Faster training and inference times are made possible by GNNs' ability to interpret graph data in a distributed, parallel fashion.

# CHAPTER 3

## METHODOLOGY

In this chapter, we present a detailed step-by-step procedure for our proposed model.

### 3.1 Algorithm for our proposed model

Suppose each item is  $Q = \{q_1, q_2, \dots, q_m\}$ . Any session consisting of consecutive interaction (i.e., item clicked by one user) in a particular order can be expressed as  $S = \{qs_1, qs_2, \dots, qs_l\}$ . The  $q_i$  item clicked in session  $S$  is  $q_{si}$ , and the number of elements in  $S$  is  $l$ .

The basic aim of a session based recommendation algorithm is to recommends the top- $N$  items ( $1 \leq N \leq |Q|$ ) from  $Q$  most likely to be completed by a user in the ongoing session  $S$ . A snapshot of our proposed model is presented in Fig. 3.2. The model is fed unprocessed datasets including interactions between users and objects. Here, we go over the specific steps we took to train our model.

- Data preprocessing: To make the datasets useful for the model, we conduct various analysis on the raw dataset and make some wise decisions.
  - Eliminating pointless sessions, such as those with a duration of 1.
  - Items with fewer than five occurrences are being removed.
  - Separating the training and testing datasets.
- Learning stage: We find out the relationship among various items and their characteristic features.
  - Each element in the outer list correspond to an item in the dataset, and the inner list shows the items that are next to that item. The adjacency list is a list of lists that describe the adjacency information. For instance, the list of neighbours for the item with ID 0 would be provided by `adj[0]`.
  - The weight list displays the weights associated with the adjacency relationships. Each entry in the weight list represents an item in the dataset, while the inner list contains the weights of the neighbouring objects. The weights indicate the frequency or strength of a relationship between items. For example, `weight[0]` would offer the list of weights for the item's neighbours with ID 0.
- Dense vectors that represent the object's features are called item embeddings. These vectors are discovered throughout the model's training phase, where the model At time-step  $t$ , or  $h_t$ , each item that belongs to  $Q$  is encoded into a unified embedding space, which is subsequently converted into a  $d$ dimensional latent vector space.

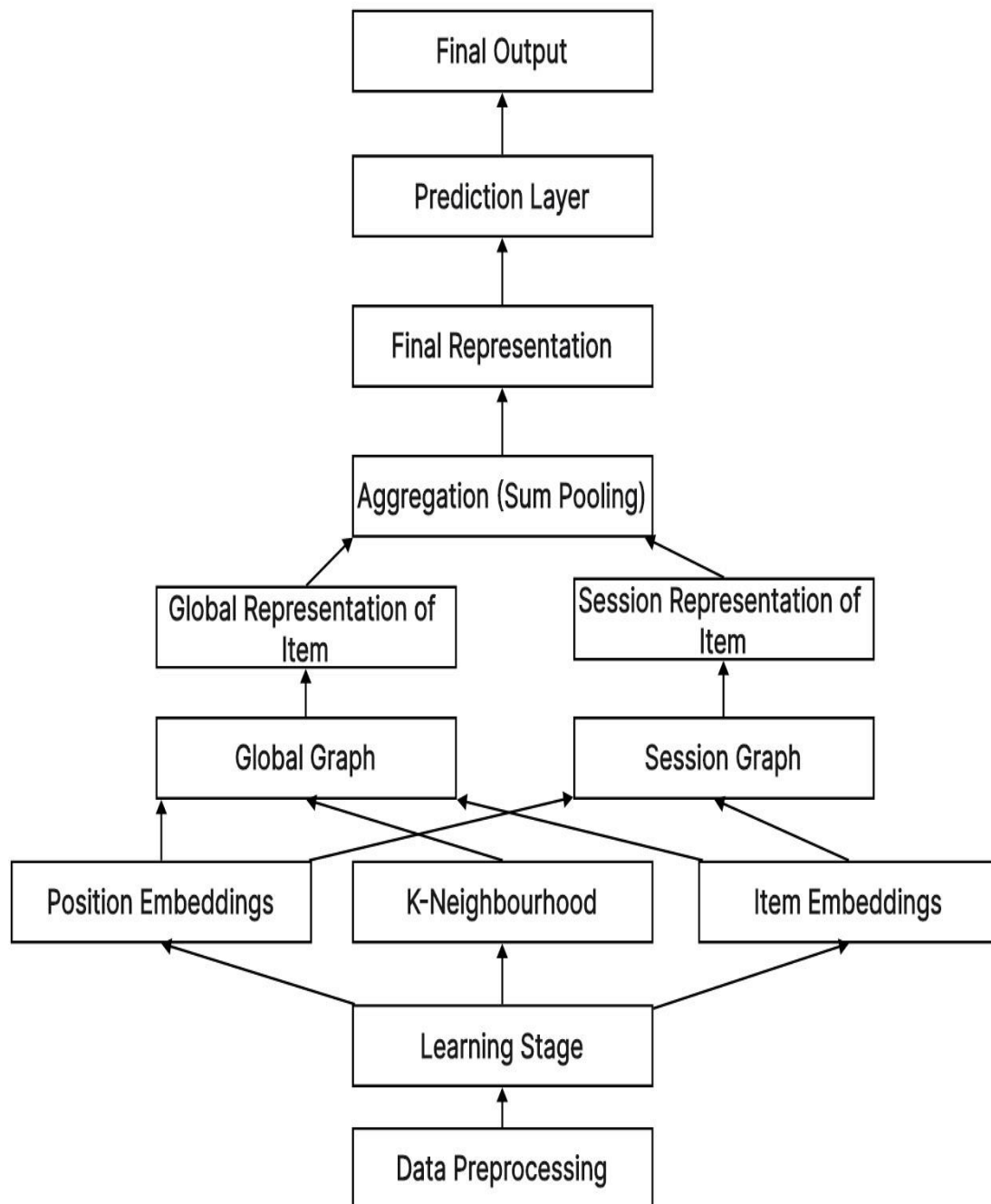


Figure 3.1: Flow chart of the proposed model tries to find meaningful representations for the items based on the available data.

- Global graph: It is the mechanism to capture the item-transitions at inter-session level.
  - The global graph's concept is to take a top-down look at all of the dataset's item transitions. This is accomplished by taking into account every pairwise item transfer that has occurred during different sessions. By connecting each item-transition that has occurred during a session, we want to build a global graph in our model. Not just the current session is taken into consideration. By doing this, we incorporate contextual information into our model while accounting for item-transitions from earlier models.

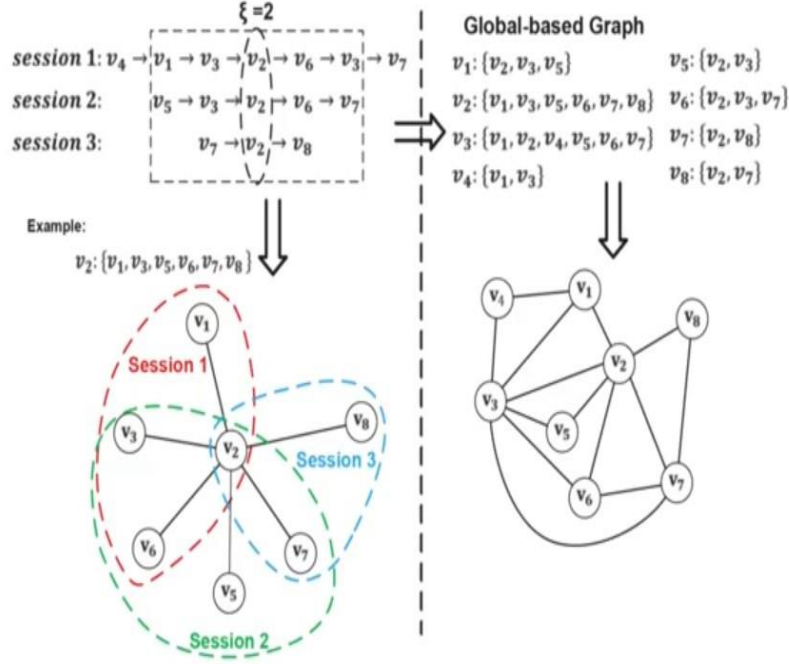


Figure 3.2: Global Graph

- We use item-KNN to determine each item's neighbourhood.  $Nk(q)$  is used to symbolise it.
- Let  $G_g = (Q_g, E_g)$  be the global graph, where  $E_g = \{e_{gij} | (q_i, q_j) | q_i \in Q, q_j \in Nk(q_i)\}$  indicates the set of edges, and  $Q_g$  is the graphs node set that contains all items in  $Q$ . Every edge represents two paired items chosen from every session. We obtain the representation of the item. We apply dropout to remove overfitting.

$$h_{g,(k)q} = \text{dropout}(h_{g,(k)q}) \quad (3.1)$$

- Session graph: It is the mechanism that take note of item transitions at intra-session level.
  - In order to comprehend the session-level embeddings of the items, the session-graph aims to describe the specific sequential pattern of the pairwise neighbouring items of the current session. Graphical neural networks (GNN) convert each session sequence into a session graph in order to learn the embeddings of the objects in a given session.
  - Let's have a session. Thus,  $G_s = \{Q_s, E_s\}$  will be used to describe the related session graph. Items that have had any action done during session  $S$ —such as a click, comment, rating, purchase, etc.—are represented by  $Q_s$ , which is a subset of  $Q$ .  $E_s$  is a collection

of eSij-shaped edges, each of which denotes a session-level pattern of transition between two neighbouring objects. A pair of endpoints  $e = \{q_i, q_j\}$  represent an edge. We have taken the concept and applied four types of edges [78, 51].

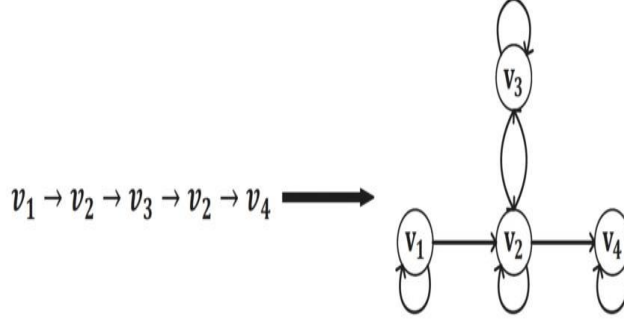


Figure 3.3: Local graph

- \*  $e_{in}$  refers to the edge from  $q_i$  to  $q_j$  representing forward transition.
  - \*  $e_{out}$  refers to the edge from  $q_j$  to  $q_i$  representing backward transition.
  - \*  $e_{dual}$  refers to the edge that has both transition between the two vertices  $q_i$  and  $q_j$ .
  - \*  $q_{self}$  refers to the edge that both ends are on the same vertex.
- The attention mechanism is used to determine the weights among different nodes in order to learn session level embedding. The following formula is used to determine the attention score.

$$e_{ij} = \text{LeakyReLU } \mathbf{a}^\top \mathbf{r}_{ij} (\mathbf{h}_{q_i} \odot \mathbf{h}_{q_j}) \quad (3.2)$$

where  $e_{ij}$  denotes the significance of the node  $q_j$ 's feature to node  $q_i$ . Nonlinearity is introduced by utilizing LeakyReLU activation function. The relation between  $q_i$  and  $q_j$  is denoted as  $\mathbf{r}_{ij}$ , and  $\mathbf{a}^* \in \mathbb{R}^d$  represents the weight vectors.

To normalize the attention weight, we use softmax function.

$$\alpha^{ij} = \frac{\exp(\text{LeakyReLU } \mathbf{a}^\top \mathbf{r}_{ij} (\mathbf{h}_{q_i} \odot \mathbf{h}_{q_j}))}{\sum_{q^k \in \mathcal{N}} \exp(\text{LeakyReLU } (\mathbf{a}^\top \mathbf{r}_{ik} (\mathbf{h}_{q_i} \odot \mathbf{h}_{q^k})))} \quad (3.3)$$

Here,  $\alpha_{ij}$  represents the normalized attention weights between node  $q_i$  and  $q_j$ . The LeakyReLU function introduces non-linearity,  $\mathbf{r}_{ij}$  denotes the relation between  $q_i$  and  $q_j$ ,  $\mathbf{a}^*$  is the weights vector, and  $\mathcal{N}_s$  represents the set of neighboring nodes for  $q_i$  within the session.

- By taking a linear combination of the feature corresponding to the coefficient, the output features for each node are calculated by following equation 3.4.

$$\mathbf{h}_{s,q_i} = \sum_{q_j \in N_s(q_i)} \alpha_{ij} \mathbf{h}_{q_j} \quad (3.4)$$

Here,  $\mathbf{h}_{s,q_i}$  represents the output features for node  $q_i$  within the session. The coefficients  $\alpha_{ij}$  are computed based on the attention weights, and  $\mathbf{h}_{q_j}$  denotes the features of node  $q_j$ . The sum is taken over the neighboring nodes  $q_j$  within the session  $N_s(q_i)$ .

• Representation of final item in the session

- The end session representation of objects is possible by integrating the session-level and global-level information derived from the global graph and, session graph respectively. Each item's immediate and wider context are reflected in a single representation that combines session-level and global-level data. Both local details and global patterns are incorporated into the final composite representation, which offers a more thorough comprehension of every item.
- We have used sum-pooling for this task as represented in equation 3.5.

$$h'_q = h_{g,(k)q} + h_s \quad (3.5)$$

- By calculating the average of the session's item representation, the session's information may be derived by following equation 3.6.

$$s' = \frac{1}{l} \sum_{i=1}^l h'_{q_{s_i}} \quad (3.6)$$

A soft-attention process is used for learning the corresponding weights:

$$\beta_i = t^T \sigma(M_4 z_i + M_5 s' + b_4) \quad (3.7)$$

The item representations  $h'_{q_{s_i}}$  weighted by the appropriate  $\beta_i$  values are linearly combined to create the session representation, denoted by  $S$  in equation 3.8. While  $t_2$  and  $b_4$  are vectors of size  $d$ , the parameters  $M_4$  and  $M_5$  are matrices of dimension  $d \times d$ . The weights  $\beta_i$  can be calculated with the use of these learnable parameters. A linear combination of the item representation  $h'_{q_{s_i}}$  for each item in the current session can be used to determine the session  $S$ 's ultimate result. The contribution of each item can be determined by using equation 3.8, which combines information from the session graph and the sequential order.

$$S = \sum_{i=1}^l \beta_i h'_{q_{s_i}} \quad (3.8)$$

Generating recommendations: In this stage, we take each item's final representation and embed it. Equation 3.9 displays the final score after we conduct a dot product. This score indicates how similar or comparable the item is to the user. A higher score mean that the item and the user's tastes are more closely aligned.

$$\hat{y}_i = \text{Softmax}(S^T h_q) \quad (3.9)$$

- Utilising the cross-entropy, the loss function for our model is constructed by following the equation 3.10.

$$L(\hat{y}) = -\sum_{i=1}^m y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \quad (3.10)$$

# CHAPTER 4

## RESULT AND EXPERIMENTAL ANALYSES

In this chapter, we are representing all the specific values for our parameters that we took to carry out our experiments.

### 4.1 Datasets used

The following datasets were utilised to test our model.

- The CIKM Cup of 2016 provided the Diginetica dataset. It includes standard transaction data.
- The IJCAI-15 competition provided the TSmall dataset. It includes session data collected by anonymous users. TSmall is an online store.
- The Nowplaying dataset contains information on the music that users have listened to.

Table I: Dataset statistics after preprocessing

Dataset	Diginetica	Tsmall	Nowplaying
Number of clicks	9,82,961	8,18,479	13,67,963
Number of items	43,097	40,728	60,417
Average length	5.12	6.69	7.42

### 4.2 Data Preprocessing

We have done the preprocessing work in accordance with [82, 83, 78].

- Sessions of a single item length were eliminated.
- We eliminated items with an overall appearance count of fewer than five across all sessions.
- We followed [43] to separate the datasets into training and test data, designating the session that occurred during the last week as test data due to their recent nature and the other data as training data.



### 4.3 Evaluation metrics

For evaluating our model against various other models, we have used the following metrics by following [43, 82, 78].

- $P@k$ : Precision@k determines the precision of the top-k recommendation. The percentage of the recommended item at positions 1 through k that are actually relevant to the user is evaluated. A greater proportion of the recommended item are relevant when the precision@k number is larger. For instance, the percentage of relevant things to the top 5 suggested items is computed when precision@5 is taken into account. With a  $P@5$  grade of 0.8, four of the top five suggestions were pertinent to the user.
- Mean Reciprocal Rank at position k is abbreviated as  $MRR@k$ . It evaluates the first pertinent item in the recommendations' ranking quality. Models that rate relevant things higher receive higher marks from  $MRR@k$ , which take into account the location of the first relevant item. Consider  $MRR@10$ , which determines the avg. reciprocal rank of the first pertinent item among the top ten suggested things. The first relevant item was typically located at position 5 in the suggestions when the  $MRR@10$  score was 0.5.
- We took  $k=10, 20$  for our evaluation process by following [78], [82].

### 4.4 Parameter setup

The parameter we used for our model are as follows: [43, 82, 37, 78].

- The Gaussian distribution is used to initialise the parameters. The mean is zero, while the standard deviation is 0.1.
- The latent vector has a size of 100.
- The starting learning rate is assumed to be 0.001 for the Adam optimiser, which we are utilising. Every third epoch, it will decay by 0.1.
- 10–5 is the penalty for L2.
- There are twelve neighbours.
- Twelve is the maximum distance allowed between neighbouring items.

### 4.5 Item-KNN Parameter

When recommending things based on how similar they are to the current item in a session, the parameter "K" in Item-KNN sets how many closest neighbours to take into account. Greater "K" values enable a more thorough examination of item similarities, which improves recommendation outcomes. In our model, we have set K to 40.

### 4.6 Baseline Algorithms

Table II lists all the algorithms with which we compare our model with.

Table II: Baseline algorithms

Technique	Research Papers
Item-KNN[62]	Depending on how closely the things in the current session connect to those in earlier sessions, it suggests items.
FPMC[56]	Matrix factorisation is integrated with the first-order Markov chain. Both users preferences and sequential impacts are taken into account.
GRU4Rec[27]	This RNN-based model simulates user sequences using Gated Recurrent Units (GRU).
NARM[37]	It is an extension of GRU4Rec[27] as it introduces attention to RNN for session based recommender system.
STAMP[43]	It employs attention layers to catch the user's momentary interest by relying on their self-attention on the preceding item in the current session.
SR-GNN[82]	It obtain item embedding via a gated GNN layer and computes session-level embeddings using selfattention.
CSRM[71]	It makes use of a memory network to take into consideration the structure of previous $n$ sessions. It helps in making a better prediction of the current session.
FGNN[52]	It uses attention weights of the graphical layers for understanding item embeddings and a feature extractor at the graphical level for making the session recommendations.

## 4.7 Result

For our experiments, we have utilised Google Collaborative. We made use of 128 GB of storage, 8 vCPUs, 16 GB of RAM, and 16 GB of GPU RAM. For 20 epochs, we applied our model to every dataset.

- Table III shows how well our suggested model performs when compared to each of the baseline models mentioned above. This comparison was conducted using  $P@20$  and  $MRR@20$  as the measures.
- Table IV shows how well our suggested model performs when compared to each of the baseline models mentioned previously. This comparison was conducted using the measures  $P@10$  and  $MRR@10$ .
- The impact of the dropout ratio on  $P@20$  for the TMall dataset is seen in Fig. 4.1.
- The trajectory of the loss function in relation to the number of epochs is shown in Fig. 4.2.
- The  $P@20$ 's trajectory with the number of epochs for the TMall dataset is shown in Fig. 4.3.

- When one of the essential elements of our model was eliminated, the outcome is displayed in Table V. The global graph and the session graph are the two essential elements.
- The outcomes of using various techniques for the crucial aggregating processes are shown in Table VI. We employed concatenation, sum pooling, max pooling, and gate mechanisms.

Table III: Results for @20

Models/Dataset	Diginetica		TMall		Nowplaying	
	P at 20	MRR at 20	P at 20	MRR at 20	P at 20	MRR at 20
Item-KNN	35.74	11.57	9.14	3.31	15.94	4.91
FPMC	22.15	6.66	16.05	7.32	7.36	2.82
GRU4REC	30.78	8.22	10.94	5.89	7.92	4.48
NARM	48.33	16.00	23.31	10.70	18.59	6.93
STAMP	46.63	15.13	26.46	13.36	17.66	6.88
CSRM	50.56	16.38	29.47	13.96	18.14	6.42
SR-GNN	51.25	17.78	27.58	13.72	18.87	7.47
FGNN	50.57	16.84	25.25	10.39	18.78	7.15
Our model	53.98	19.02	32.05	15.05	21.30	8.47

Table IV: Results for at 10

Models/Dataset	Diginetica		TMall		Nowplaying	
	P at 10	MRR at 10	P at 10	MRR at 10	P at 10	MRR at 10
Item-KNN	25.07	10.77	6.65	3.11	10.96	4.55
FPMC	15.43	6.20	13.10	7.12	5.28	2.68
GRU4Rec	17.93	7.73	9.47	5.78	6.74	4.40
NARM	35.44	15.13	19.17	10.42	13.6	6.62
STAMP	33.98	14.26	22.63	13.12	13.22	6.57
CSRM	36.59	15.41	24.54	13.62	13.20	6.08
SR-GNN	38.42	16.89	23.41	13.45	14.17	7.15
FGNN	37.72	15.95	20.67	10.07	13.89	6.8
Our model	40.83	17.77	27.06	14.71	16.41	8.13

Table V: Performance of our model when key components were removed on a single basis

Models/Dataset	Diginetica		TMall		Nowplaying	
	P at 10	MRR at 10	P at 10	MRR at 10	P at 10	MRR at 10
W/o global graph	53.11	18.77	31.44	14.54	20.33	7.43
W/o session graph	51.78	16.21	31.22	12.67	18.11	6.55

Table VI: Effect of different aggregation operation.

Models/Dataset	Diginetica		Tmall		Nowplaying	
Measures	P at 20	MRR at 20	P at 20	MRR at 20	P at 20	MRR at 20
Gate Mechanism	52.34	18.10	32.80	15.33	22.47	7.83
Max Pooling	45.39	16.45	31.87	15.39	19.13	6.71
Concatenation	50.22	17.02	31.55	14.89	19.88	7.93
Sum Pooling	53.97	19.01	32.04	15.05	21.30	8.44

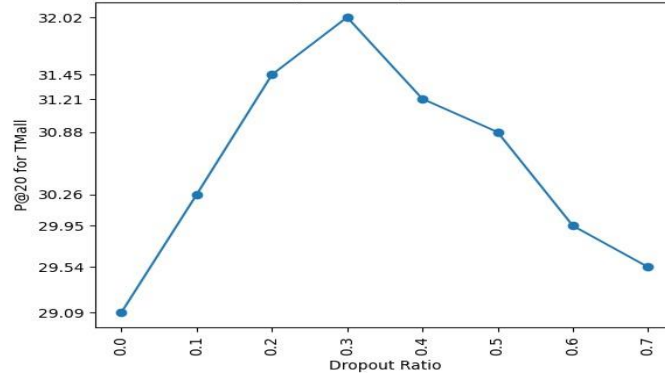


Figure 4.1: Impact of Dropout ratio

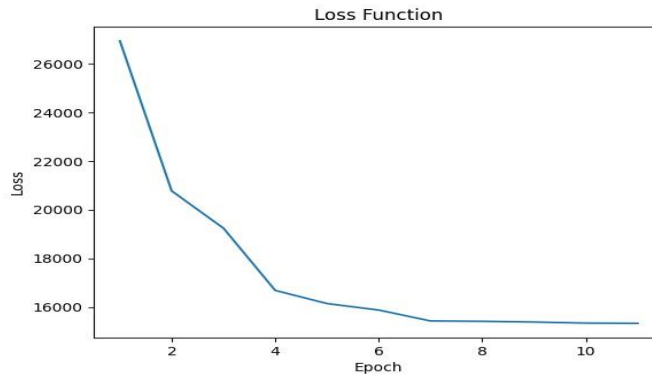


Figure 4.2: Stabilization of loss function with epoch for Tmall dataset

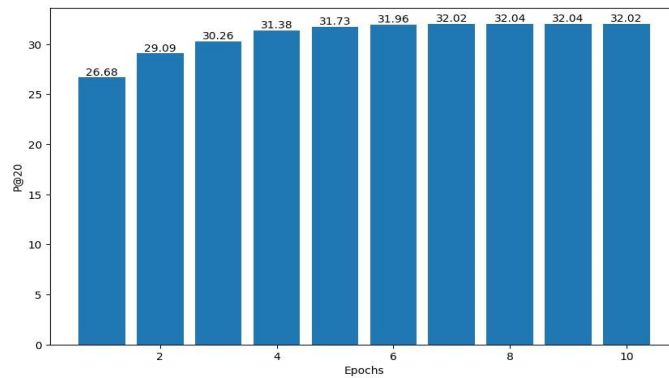


Figure 4.3: Trajectory of P@20 vs epochs for Tmall dataset

## 4.8 Observations

With our experiments, we have observed the following conclusions-

- It is evident from the data in Tables III and IV that our model has produced superior outcomes when compared to models that do not make use of graphical neural networks. This demonstrates unequivocally the great success of graphical neural network in the field of session-based recommendation systems
- Our model even outperformed all of the graphical neural network-based session-based recommender systems we used for our studies, according to the data shown in Tables III and IV.
- By the time it reaches the tenth epoch, as shown in Fig. 4.1, our model is stabilised. It is evident from Fig. 4.3 that the TMall dataset's peak performance for P@20 was attained in the ninth epoch. This also applies to other datasets.
- The information included in Table V indicates the impact of eliminating either of the two essential elements of our model. It can be concluded that while the removal of session graphs has a significant impact on suggestions, the removal of the global graph has no discernible effect.
- It is evident that the data in Table VI that the sum-pooling technique perform best when it comes to combining session and global item information. Additionally, the gate mechanism performed well and, in one case, exceeded sum-pooling (P@20 for Nowplaying).
- Fig. 4.1 shows the impact of the dropout rate. For the global graph representation, we employed the regularisation method known as dropout. Using all neurones for testing and randomly discarding specific neurones with probability  $p$  during training is the fundamental idea behind dropout. Since it is easy to overfit, we may observe that the model performs poorly on both datasets when the dropout ratio is low. The dropout ratio functions best when it falls between the extremes.

The model's performance starts to suffer when the dropout ratio reaches the higher extreme end since it is difficult for the model is to learn from the data when there aren't enough neurones available.

# CONCLUSION

Our report's conclusion concentrated on the study of recommender systems, specifically session-based recommender systems. We investigated different methods and models used in this field during our study and assessed how well they performed in comparison to baseline models.

Our study's key finding is that, in terms of recommendation accuracy, our Graph Neural Networks (GNN)-based model performed better than all of the baseline models. In order to produce more precise and tailored suggestions, the GNN model demonstrated exceptional performance in identifying the intricate correlations and patterns present in session-based data.

Our model effectively integrated the sequential and contextual information found in session data by utilising GNNs, which enables it to adjust to user preferences and provide accurate recommendations based on specific sessions. Our model's increased personalisation and accuracy have great potential to improve user experience and engagement in practical recommendation scenarios.

We can state with confidence that our GNN-based model has proven its effectiveness and potential in session-based recommender systems after a thorough evaluation procedure and comparison with several baseline models. All things considered, our results add to the expanding corpus of research on recommender systems, particularly as it relates to session based recommendations. Our GNN model's performance emphasises how crucial it is to use cutting-edge methods to successfully address the difficulties posed by session-based data. To sum up, our study of session based recommender systems, especially our GNN-based model, demonstrates the enormous potential for enhancing suggestion personalisation and accuracy. We believe that further research and development of GNN-based techniques will advance recommender systems and enhance user experience in a number of domains.

# REFERENCES

- [1] Amr Abdelhamed and Hongning Wang. “S3rec: Self-supervised learning for sequential recommendation”. In: *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM)*. 2020.
- [2] Gediminas Adomavicius and YoungOk Kwon. “Improving aggregate recommendation diversity using ranking-based techniques”. In: *IEEE Transactions on Knowledge and Data Engineering* 24.5 (2011), pp. 896–911.
- [3] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. “Mining association rules’ between sets of items in large databases”. In: *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*. 1993, pp. 207–216.
- [4] Sameer Anand et al. “Integrating node centralities, similarity measures, and machine learning classifiers for link prediction”. In: *Multimedia tools and applications* 81.27 (2022), pp. 38593–38621.
- [5] Chumki Basu. “Recommendation as Classification: Using Social and Content-Based Information in Recommendation Chumki Basu”. In: (1998).
- [6] Robert M Bell and Yehuda Koren. “Scalable collaborative filtering with jointly derived neighborhood interpolation weights”. In: *Seventh IEEE international conference on data mining (ICDM 2007)*. IEEE. 2007, pp. 43–52.
- [7] Neeraj Bhat, Navneet Saggu, Sanjay Kumar, et al. “Generating visible spectrum images from thermal infrared using conditional generative adversarial networks”. In: *2020 5th International Conference on Communication and Electronics Systems (ICCES)*. IEEE. 2020, pp. 1390–1394.
- [8] Sergey Brin, Rajeev Motwani, and Craig Silverstein. “Beyond market baskets: Generalizing association rules to correlations”. In: *Proceedings of the 1997 ACM SIGMOD international conference on Management of data*. 1997, pp. 265–276.
- [9] Robin Burke. “Hybrid recommender systems: Survey and experiments”. In: *User modeling and user-adapted interaction* 12 (2002), pp. 331–370.
- [10] Robin Burke. “Hybrid web recommender systems”. In: *The adaptive web: methods and strategies of web personalization* (2007), pp. 377–408.
- [11] Shi-Yong Chen et al. “Stabilizing reinforcement learning in dynamic environment with application to online recommendation”. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2018, pp. 1187–1196.
- [12] Xu Chen et al. “Personalized key frame recommendation”. In: *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*. 2017, pp. 315–324.
- [13] Mark Claypool et al. “Combing content-based and collaborative filters in an online newspaper”. In: *Proc. of Workshop on Recommender Systems-Implementation and Evaluation*. 1999.

- [14] Paul Covington, Jay Adams, and Emre Sargin. “Deep neural networks for youtube recommendations”. In: *Proceedings of the 10th ACM conference on recommender systems*. 2016, pp. 191–198.
- [15] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. “Are we really making much progress? a worrying analysis of recent neural recommendation approaches”. In: *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys)*. 2019.
- [16] Mostafa Dehghani et al. “Neural personalized ranking for item cold-start recommendation”. In: *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 2018.
- [17] Shuiguang Deng et al. “On deep learning for trust-aware recommendations in social networks”. In: *IEEE transactions on neural networks and learning systems* 28.5 (2016), pp. 1164–1177.
- [18] C Desrosiers and G Karypis. “Methods, A comprehensive survey of neighborhoodbased recommendation”. In: (2011).
- [19] Xin Dong et al. “A hybrid collaborative filtering model with deep structure for recommender systems”. In: *Proceedings of the AAAI Conference on artificial intelligence*. Vol. 31. 1. 2017.
- [20] Travis Ebesu and Yi Fang. “Neural citation network for context-aware citation recommendation”. In: *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*. 2017, pp. 1093–1096.
- [21] Michael D Ekstrand, John T Riedl, Joseph A Konstan, et al. “Collaborative filtering recommender systems”. In: *Foundations and Trends® in Human–Computer Interaction* 4.2 (2011), pp. 81–173.
- [22] Nir Friedman, Dan Geiger, and Moises Goldszmidt. “Bayesian network classifiers”. In: *Machine learning* 29 (1997), pp. 131–163.
- [23] Yuyun Gong and Qi Zhang. “Hashtag recommendation using attention-based convolutional neural network.” In: *IJCAI*. 2016, pp. 2782–2788.
- [24] Will Hamilton, Zhitao Ying, and Jure Leskovec. “Inductive representation learning on large graphs”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)*. 2017.
- [25] Jiawei Han, Jian Pei, and Yiwen Yin. “Mining frequent patterns without candidate generation”. In: *ACM sigmod record* 29.2 (2000), pp. 1–12.
- [26] Balázs Hidasi et al. “Session-based recommendations with recurrent neural networks”. In: *Proceedings of the 4th International Conference on Learning Representations (ICLR)*. 2015.
- [27] Balázs Hidasi et al. “Session-based recommendations with recurrent neural networks”. In: *arXiv preprint arXiv:1511.06939* (2015).
- [28] Yogesh Jhamb, Travis Ebesu, and Yi Fang. “Attentive contextual denoising autoencoder for recommendation”. In: *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*. 2018, pp. 27–34.



- [29] Wang-Cheng Kang and Julian McAuley. “Self-attentive sequential recommendation”. In: *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys)*. 2018.
- [30] Thomas N Kipf and Max Welling. “Semi-supervised classification with graph convolutional networks”. In: *arXiv preprint arXiv:1609.02907* (2016).
- [31] Yehuda Koren. “Factorization meets the neighborhood: a multifaceted collaborative filtering model”. In: *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2008, pp. 426–434.
- [32] Yehuda Koren, Robert Bell, and Chris Volinsky. “Matrix factorization techniques for recommender systems”. In: *Computer* 42.8 (2009), pp. 30–37.
- [33] Akshi Kumar, Nipun Aggarwal, and Sanjay Kumar. “SIRA: a model for propagation and rumor control with epidemic spreading and immunization for healthcare 5.0”. In: *Soft Computing* 27.7 (2023), pp. 4307–4320.
- [34] Sanjay Kumar et al. “Movie genre classification using binary relevance, label powerset, and machine learning classifiers”. In: *Multimedia Tools and Applications* 82.1 (2023), pp. 945–968.
- [35] Sanjay Kumar et al. “OptNet-Fake: Fake News Detection in Socio-Cyber Platforms Using Grasshopper Optimization and Deep Neural Network”. In: *IEEE Transactions on Computational Social Systems* (2023).
- [36] Urszula Kuzelewska. “Advantages of information granulation in clustering algorithms”. In: *Agents and Artificial Intelligence: Third International Conference, ICAART 2011, Rome, Italy, January, 28-30, 2011. Revised Selected Papers 3*. Springer. 2013, pp. 131–145.
- [37] Jing Li et al. “Neural attentive session-based recommendation”. In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 2017, pp. 1419–1428.
- [38] Sheng Li, Jaya Kawale, and Yun Fu. “Deep collaborative filtering via marginalized denoising auto-encoder”. In: *Proceedings of the 24th ACM international on conference on information and knowledge management*. 2015, pp. 811–820.
- [39] Xiaopeng Li and James She. “Collaborative variational autoencoder for recommender systems”. In: *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 2017, pp. 305–314.
- [40] Yang Li et al. “Hashtag recommendation with topical attention-based LSTM”. In: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 2016, pp. 3019–3029.
- [41] Yujia Li et al. “Gated graph sequence neural networks”. In: *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. 2015.
- [42] Defu Lian et al. “GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation”. In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2014, pp. 831–840.

- [43] Qiao Liu et al. “STAMP: short-term attention/memory priority model for sessionbased recommendation”. In: *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 2018, pp. 1831–1839.
- [44] Abhishek Mallik, Anavi Khetarpal, and Sanjay Kumar. “ConRec: malware classification using convolutional recurrence”. In: *Journal of Computer Virology and Hacking Techniques* 18.4 (2022), pp. 297–313.
- [45] Benjamin M Marlin. “Modeling user rating profiles for collaborative filtering”. In: *Advances in neural information processing systems* 16 (2003).
- [46] Prem Melville and Vikas Sindhwani. “Recommender systems.” In: *Encyclopedia of machine learning* 1 (2010), pp. 829–838.
- [47] Aditya Miglani et al. “A Literature Review on Brain Tumor Detection and Segmentation”. In: *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE. 2021, pp. 1513–1519.
- [48] Hanh TH Nguyen et al. “Personalized deep learning for tag recommendation”. In: *Advances in Knowledge Discovery and Data Mining: 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part I* 21. Springer. 2017, pp. 186–197.
- [49] Deuk Hee Park et al. “A literature review and classification of recommender systems research”. In: *Expert systems with applications* 39.11 (2012), pp. 10059– 10072.
- [50] Michael J Pazzani and Daniel Billsus. “Content-based recommendation systems”. In: *The adaptive web: methods and strategies of web personalization* (2007), pp. 325–341.
- [51] Ruihong Qiu et al. “Rethinking the item order in session-based recommendation with graph neural networks”. In: *Proceedings of the 28th ACM international conference on information and knowledge management*. 2019, pp. 579–588.
- [52] Ruihong Qiu et al. “Rethinking the item order in session-based recommendation with graph neural networks”. In: *Proceedings of the 28th ACM international conference on information and knowledge management*. 2019, pp. 579–588.
- [53] Massimo Quadrana et al. “Personalizing session-based recommendations with hierarchical recurrent neural networks”. In: *Proceedings of the 11th ACM Conference on Recommender Systems (RecSys)*. 2017.
- [54] Steffen Rendle. “Factorization machines”. In: *2010 IEEE International conference on data mining*. IEEE. 2010, pp. 995–1000.
- [55] Steffen Rendle. “Factorization machines with libfm”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 3.3 (2012), pp. 1–22.
- [56] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. “Factorizing personalized markov chains for next-basket recommendation”. In: *Proceedings of the 19th international conference on World wide web*. 2010, pp. 811–820.
- [57] Steffen Rendle et al. “Factorizing personalized markov chains for next-basket recommendation”. In: *Proceedings of the 19th International Conference on World Wide Web (WWW)*. 2010.

- [58] Yash Saini et al. “Abusive text examination using Latent Dirichlet allocation, self organizing maps and k means clustering”. In: *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE. 2020, pp. 1233– 1238.
- [59] Yash Saini et al. “Abusive text examination using Latent Dirichlet allocation, self organizing maps and k means clustering”. In: *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE. 2020, pp. 1233– 1238.
- [60] Ruslan Salakhutdinov and Andriy Mnih. “Bayesian probabilistic matrix factorization using Markov chain Monte Carlo”. In: *Proceedings of the 25th international conference on Machine learning*. 2008, pp. 880–887.
- [61] Badrul Sarwar et al. “Item-based collaborative filtering recommendation algorithms”. In: *Proceedings of the 10th international conference on World Wide Web*. 2001, pp. 285–295.
- [62] Badrul Sarwar et al. “Item-based collaborative filtering recommendation algorithms”. In: *Proceedings of the 10th international conference on World Wide Web*. 2001, pp. 285–295.
- [63] Sandeep Singh Sengar et al. “Bot detection in social networks based on multilayered deep learning approach”. In: *Sensors & Transducers* 244.5 (2020), pp. 37– 43.
- [64] Sandeep Singh Sengar et al. “Bot detection in social networks based on multilayered deep learning approach”. In: *Sensors & Transducers* 244.5 (2020), pp. 37– 43.
- [65] Barry Smyth and Paul Cotter. “A personalised TV listings service for the digital TV age”. In: *Knowledge-Based Systems* 13.2-3 (2000), pp. 53–59.
- [66] Yong Kiam Tan, Xinxing Xu, and Yong Liu. “Improved recurrent neural networks for session-based recommendations”. In: *Proceedings of the 1st workshop on deep learning for recommender systems*. 2016, pp. 17–22.
- [67] Jiayi Tang et al. “Personalized top-n sequential recommendation via convolutional sequence embedding”. In: *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys)*. 2018.
- [68] Trinh Xuan Tuan and Tu Minh Phuong. “3D convolutional networks for sessionbased recommendation with content features”. In: *Proceedings of the eleventh ACM conference on recommender systems*. 2017, pp. 138–146.
- [69] Bartłomiej Twardowski. “Modelling contextual information in session-aware recommender systems with neural networks”. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. 2016, pp. 273–276.
- [70] Jun Wang et al. “Irgan: A minimax game for unifying generative and discriminative information retrieval models”. In: *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 2017, pp. 515– 524.
- [71] Meirui Wang et al. “A collaborative session-based recommendation approach with parallel memory modules”. In: *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*. 2019, pp. 345– 354.

- [72] Qinyong Wang et al. “Neural memory streaming recommender networks with adversarial training”. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2018, pp. 2467–2475.
- [73] Shoujin Wang et al. *A Survey on Session-based Recommender Systems*. 2021. arXiv: 1902.04864[cs.IR].
- [74] Suhan Wang et al. “What your images reveal: Exploiting visual contents for pointof-interest recommendation”. In: *Proceedings of the 26th international conference on world wide web*. 2017, pp. 391–400.
- [75] Xiang Wang et al. “Item silk road: Recommending items from information domains to social users”. In: *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 2017, pp. 185–194.
- [76] Xinxi Wang and Ye Wang. “Improving content-based and hybrid music recommendation using deep learning”. In: *Proceedings of the 22nd ACM international conference on Multimedia*. 2014, pp. 627–636.
- [77] Xinxi Wang et al. “Exploration in interactive personalized music recommendation: a reinforcement learning approach”. In: *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 11.1 (2014), pp. 1–22.
- [78] Ziyang Wang et al. “Global Context Enhanced Graph Neural Networks for SessionBased 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. 169–178. ISBN: 9781450380164. DOI: 10.1145/3397271.3401142. URL: <https://doi.org/10.1145/3397271.3401142>.
- [79] Caihua Wu et al. “Recurrent neural network based recommendation for time heterogeneous feedback”. In: *Knowledge-Based Systems* 109 (2016), pp. 90–103.
- [80] Chao-Yuan Wu et al. “Recurrent recommender networks”. In: *Proceedings of the tenth ACM international conference on web search and data mining*. 2017, pp. 495–503.
- [81] Shu Wu et al. “Session-based recommendation with graph neural networks”. In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*. 2019.
- [82] Shu Wu et al. “Session-based recommendation with graph neural networks”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01. 2019, pp. 346–353.
- [83] Chengfeng Xu et al. “Graph Contextualized Self-Attention Network for Sessionbased Recommendation.” In: *IJCAI*. Vol. 19. 2019, pp. 3940–3946.
- [84] Carl Yang et al. “Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation”. In: *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 2017, pp. 1245–1254.
- [85] Fajie Yuan et al. “Simple and effective next item recommendation algorithms for sequential data”. In: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 2019.

- [86] Chen Zhang et al. “SSUM: Self-Supervised User Embedding for Session-based Recommendation”. In: *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM)*. 2021.
- [87] Fuzheng Zhang et al. “Collaborative knowledge base embedding for recommender systems”. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016, pp. 353–362.
- [88] Shuai Zhang et al. “Deep learning based recommender system: A survey and new perspectives”. In: *ACM computing surveys (CSUR)* 52.1 (2019), pp. 1–38.
- [89] Yongfeng Zhang et al. “Joint representation learning for top-n recommendation with heterogeneous information sources”. In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 2017, pp. 1449–1458.
- [90] Lei Zheng, Vahid Noroozi, and Philip S Yu. “Joint deep modeling of users and items using reviews for recommendation”. In: *Proceedings of the tenth ACM international conference on web search and data mining*. 2017, pp. 425–434.
- [91] Wentao Zhu et al. “Next item recommendation with self-attention”. In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM)*. 2018.





# DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daultapur, Main Bawana Road, Delhi-42

## PLAGIARISM VERIFICATION

Title of the Thesis Design and Development of Recommendation  
System using graph neural network

Total Pages 45 Name of the Scholar Vijay Kumar

Supervisor (s)

(1) Prof. Ruchika Malhotra

(2) \_\_\_\_\_

(3) \_\_\_\_\_

Department Software Engineering

This is to report that the above thesis was scanned for similarity detection. Process and outcome is given below:

Software used: Turnitin Similarity Index: 13%, Total Word Count: 12,201

Date: 22-05-25

Vijay Kumar


Candidate's Signature

Ruchika Malhotra

Signature of Supervisor(s)



# vj thesis.docx

 Delhi Technological University

## Document Details

Submission ID

trn:oid::27535:97161445

45 Pages

Submission Date

May 22, 2025, 1:01 PM GMT+5:30

12,201 Words

Download Date

May 22, 2025, 1:08 PM GMT+5:30

64,878 Characters

File Name

vj thises3.docx

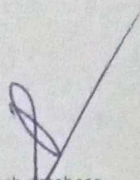
File Size

761.5 KB



# 13% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.






## Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 8 words)

## Match Groups


- 102 Not Cited or Quoted 13%  
Matches with neither in-text citation nor quotation marks
- 0 Missing Quotations 0%  
Matches that are still very similar to source material
- 0 Missing Citation 0%  
Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%  
Matches with in-text citation present, but no quotation marks

## Top Sources

- 9%  Internet sources
- 7%  Publications
- 6%  Submitted works (Student Papers)

## Integrity Flags

### 1 Integrity Flag for Review


-  **Replaced Characters**  
9 suspect characters on 3 pages  
Letters are swapped with similar characters from another alphabet.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.



# **vj thesis.docx**

 Delhi Technological University

## **Document Details**

Submission ID

trn:oid::27535:97161445

**45 Pages**

Submission Date

May 22, 2025, 1:01 PM GMT+5:30

**12,201 Words**

**64,878 Characters**

Download Date

May 22, 2025, 1:08 PM GMT+5:30

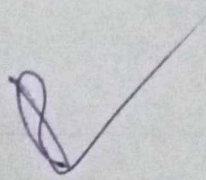
File Name

**vj thises3.docx**

File Size

**761.5 KB**







## 0% detected as AI

The percentage indicates the combined amount of likely AI-generated text as well as likely AI-generated text that was also likely AI-paraphrased.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

### Detection Groups

-  **0 AI-generated only 0%**  
Likely AI-generated text from a large-language model.
-  **0 AI-generated text that was AI-paraphrased 0%**  
Likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

#### Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (it may misidentify writing that is likely AI generated as AI generated and AI paraphrased or likely AI generated and AI paraphrased writing as only AI generated) so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

### Frequently Asked Questions

#### How should I interpret Turnitin's AI writing percentage and false positives?

The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (\*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

#### What does 'qualifying text' mean?

Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

