Optimizing Recommender systems using Graph Neural Networks

A Final Project End Semester Report

submitted by

KONDA NITIN REDDY (CS21B2028)

in partial fulfilment of requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

Department of Computer Science and Engineering
INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, DESIGN AND MANUFACTURING, KANCHEEPURAM

May 2025

DECLARATION OF ORIGINALITY

I, **KONDA NITIN REDDY**, with Roll No: **CS21B2028** hereby declare that the material presented in the Project Report titled **Optimizing Recommender systems using Graph Neural Networks** represents original work carried out by me in the **Department of Computer Science and Engineering** at the Indian Institute of Information Technology, Design and Manufacturing, Kancheepuram.

With my signature, I certify that:

- I have not manipulated any of the data or results.
- I have not committed any plagiarism of intellectual property. I have clearly indicated and referenced the contributions of others.
- I have explicitly acknowledged all collaborative research and discussions.
- I have understood that any false claim will result in severe disciplinary action.
- I have understood that the work may be screened for any form of academic misconduct.

KONDA NITIN REDDY

Place: Chennai

Date: 05.05.2025

CERTIFICATE

This is to certify that the report titled **Optimizing Recommender systems using Graph Neural Networks** , submitted by **KONDA NITIN REDDY (CS21B2028)**, to the Indian Institute of Information Technology, Design and Manufacturing Kancheepuram, in partial fulfilment of requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** is a bonafide record of the work done by him/her under my supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Umarani Jayaraman
Project Internal Guide
Assistant Professor
Department of Computer Science and Engineering
IIITDM Kancheepuram, Chennai - 600 127

Place: Chennai

Date:

ACKNOWLEDGEMENTS

I am deeply grateful to the management of Indian Institute of Information Technology Design and Manufacturing, Kancheepuram (IIITDM-K) for providing me with the opportunity to undertake this project.

I am immensely thankful to my project supervisor, Dr. Umarani Jayaraman, for her unwavering support and encouragement. Her leadership and vision have greatly inspired me to strive for excellence. Her dedication to academic excellence and innovation has been a constant source of motivation for me. Her expertise and advice have been crucial in shaping the direction of my project. Her constructive feedback and encouragement have been invaluable in overcoming challenges and achieving my project goals. I appreciate her efforts in creating an environment that nurtures creativity and critical thinking.

I would also like to thank my research group members and colleagues, including Dhanush, Prashamsa, and many others, for their collaboration and encouragement throughout this journey

Thank you all for your contributions and support.

NITIN REDDY KONDA

ABSTRACT

As digital interactions grow in volume and complexity, optimizing recommender systems has become more important than ever. Traditional methods like collaborative filtering and content-based recommendations have been useful in suggesting content based on past user behavior or item features. However, they often fall short when user preferences shift over time or in response to trends and events. These models also face issues like the cold-start problem, data sparsity, and the incapacity to capture changing relationships.

To overcome these limitations, this project introduces a recommendation system built on Temporal Graph Neural Networks (TGNNs). In contrast to static models, TGNNs depict users and objects as nodes in a dynamic graph, with time-sensitive edges representing interactions (like clicks or purchases). By incorporating the temporal aspect of user-item interactions, the model can adapt to recent user activity while still considering long-term preferences.

Our TGNN-based approach uses techniques like temporal attention, recency weighting, and memory decay to prioritize relevant information. This makes the system more responsive to changes in user behavior and improves recommendation accuracy. Designed for scalability and real-time performance, the model efficiently handles complex, high-dimensional interaction data using the strengths of graph-based learning.

KEYWORDS: Temporal Graph Neural Networks; Recommender Systems; User-Item Interaction; Adaptive Recommendations.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

AI	Artificial Intelligence
TGNN	Temporal Graph Neural Network
GNN	Graph Neural Network
TGN	Temporal Graph Network
CSV	Comma-Separated Values
JSON	JavaScript Object Notation
UI	User Interface
UX	User Experience
RDBMS	Relational Database Management System
GAT	Graph Attention Network
GCN	Graph Convolutional Network
API	Application Programming Interface
CRUD	Create, Read, Update, Delete
BPR	Bayesian Personalized Ranking
GRU	Gated Recurrent Unit

NOTATION

$G = (V, E)$	Graph with nodes V and edges E
$N(u)$	Neighborhood of node u
$h_u^{(l)}$	Node embedding at layer l for node u
W_l	Weight matrix at layer l
σ	Activation function
y_i	Actual rating of item i
$\hat{y}_i$	Predicted rating of item i
λ	Regularization parameter
$\mathcal{L}_{MSE}$	Mean Squared Error loss function
t	Timestamp of an interaction
$e_{u,i}^t$	Edge from user u to item i at time t
$\mathcal{L}_{BPR}$	Bayesian Personalized Ranking loss
d	Dimensionality of node embeddings
k	Number of top items to recommend (Top-k)
τ	Temporal decay factor

CHAPTER 1

INTRODUCTION

1.1 Background

By examining user preferences and behavior, recommender systems have become a crucial part of many online platforms, assisting consumers in finding pertinent products, services, and content. Traditionally, these systems relied on collaborative filtering and content-based methods, which model user preferences based on past interactions or item characteristics.These approaches frequently face difficulties such data sparsity, cold-start issues, and an inability to record intricate interactions between people and items, despite their effectiveness in many situations. In order to overcome these constraints, Graph Neural Networks (GNNs), a potent technology recently developed in machine learning, have been introduced. In a graph, GNNs can depict individuals and objects as nodes, while edges reflect interactions like clicks, reviews, and sales. This allows GNNs to capture both direct and higher-order relationships between users and items, enabling a more holistic understanding of user behavior. Temporal GNNs (TGNNs) extend this concept further by incorporating the time dimension, allowing for the modeling of how user-item interactions evolve over time. This capability is crucial for optimizing recommendations in dynamic environments where preferences change frequently.

1.2 Motivation

The rapid growth of digital platforms has made it increasingly important to deliver accurate, personalized recommendations in real time. Users' preferences are not static; they evolve with time as they interact with new items or experience changes in their tastes, interests, or circumstances. Traditional recommender systems fail to adequately

account for these temporal dynamics, often resulting in outdated or irrelevant recommendations. By simulating the temporal evolution of user-item interactions, TGNNs present a convincing solution to this issue and can yield more pertinent and timely recommendations. Optimizing these systems is crucial for improving user satisfaction, increasing engagement, and driving revenue on platforms that rely on accurate recommendations. By developing an optimized recommendation system using Temporal Graph Neural Networks, we aim to enhance the ability to capture both the structure and evolution of user-item interactions, offering a significant leap over conventional recommendation techniques.

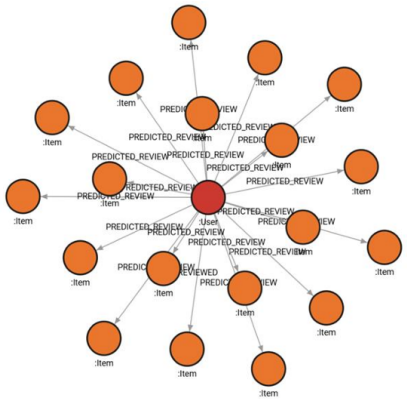


Figure 1.1: GNN architecture recommending individual user.

1.3 Problem Statement

In order to customize user experiences on a variety of digital platforms, including e-commerce and content streaming, recommender systems are essential. Traditional techniques like collaborative filtering and content-based models struggle in dynamic environments where user preferences and item characteristics change over time. These models often fail to capture the temporal nature of user-item interactions, leading to outdated recommendations. Additionally, issues like data sparsity and cold-start problems further hinder performance, especially when new users or items are involved. Temporal Graph Neural Networks (TGNNs) offer a promising solution by modeling interactions as dynamic graphs, allowing them to capture evolving user preferences and item trends. However, existing TGNN models face challenges related to scalability, computational efficiency, and optimization for real-time recommendations. This project seeks to de-

velop an optimized TGNN-based recommendation system that incorporates temporal features, improves scalability, addresses cold-start issues, and ensures high accuracy in dynamic environments.

1.4 Scope of the Project

The focus of the project is mainly on developing an optimized recommendation system using Temporal Graph Neural Networks (TGNNs). The primary objectives include:

- **Modeling User-Item Interactions:** Constructing a dynamic graph with people and objects as nodes and their interactions (clicks, ratings, and purchases) as evolving edges.
- **Capturing Temporal Dynamics:** Incorporating temporal information into the graph structure to capture how user preferences and item popularity change over time, allowing for the modeling of short-term and long term user behaviors.
- **Optimization:** Designing the TGNN architecture to ensure efficient computational performance while maintaining high accuracy in recommendations. This includes fine-tuning parameters and leveraging advanced GNN techniques like attention mechanisms, temporal decay, or memory modules.
- **Evaluation:** Comparing the TGNN-based recommendation system against traditional methods such as collaborative filtering and static GNNs to demonstrate improvements in performance metrics, such as precision, recall, and user satisfaction.
- **Applications:** The optimized TGNN model can be applied to various industries, including ecommerce, media streaming, and social media platforms, where dynamic and personalized recommendations are crucial for user engagement.

1.5 Organization of the Report

This report is structured as follows:

CHAPTER 1: INTRODUCTION An overview of recommender systems, highlighting challenges in traditional approaches and the advantages of using Temporal Graph Neural Networks (TGNNs).

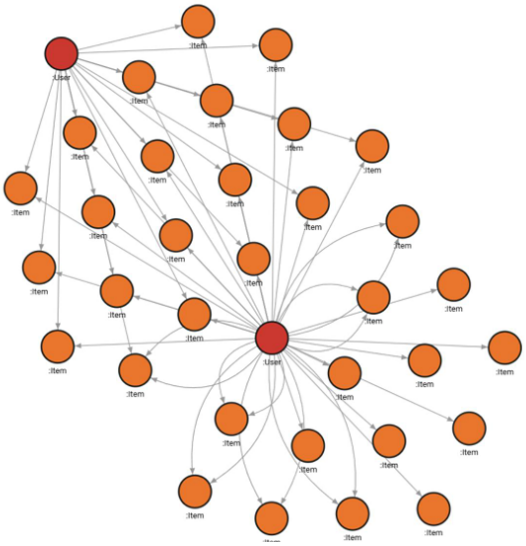


Figure 1.2: GNN architecture recommending between multi users.

CHAPTER 2: LITERATURE SURVEY A review of existing recommender system techniques, including collaborative filtering, content-based filtering, and graph neural networks. It also discusses prior research on TGNNs and identifies gaps addressed in this study.

CHAPTER 3: METHODOLOGY Details on the approach taken to develop the TGNN-based recommendation system, including dataset selection, preprocessing, model architecture, and optimization strategies.

CHAPTER 4: EXPERIMENTAL RESULTS A presentation of the system's performance, showcasing precision, recall, and accuracy. Comparisons with traditional models and visualizations of the results are also provided.

CHAPTER 5: CONCLUSIONS & FUTURE SCOPE Summarizes key findings, discusses the potential impact of the TGNN-based recommender system, and explores future enhancements, including explainability techniques and real-world deployment.

CHAPTER 2

Literature Review

2.1 Recommendation systems using graph neural networks (GNNs)

Graph neural networks, or GNNs, have become a major advancement in recommendation systems due to their ability to capture user-item interactions in an organized way. Unlike traditional approaches like collaborative filtering (CF) and matrix factorization, GNNs excel at capturing complex relationships and dependencies beyond first-order connections. Prominent early works such as **PinSage** [1] and LightGCN [2] showcased the effectiveness of GNNs in mitigating data sparsity issues and improving recommendation accuracy.

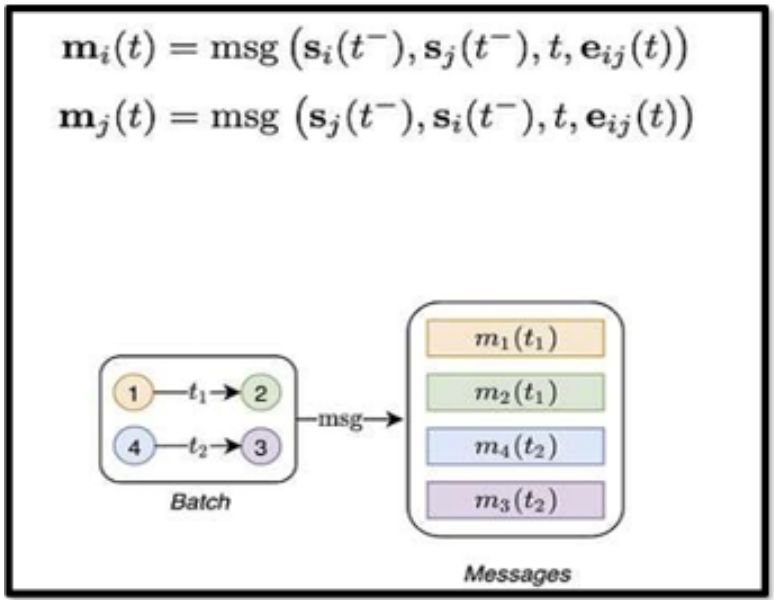


Figure 2.1: GNN Architecture

PinSage [1] pioneered the combination of random walks and GNNs to create efficient node embeddings, enabling large-scale recommendations through neighbor sampling. On the other hand, LightGCN [2] refined GNN architectures by eliminating non-linear activation functions, leading to performance gains in recommendation tasks.

Despite these developments, conventional GNN-based methods are fundamentally static and cannot adapt to the changing preferences of users over time. This limitation led to the development of Temporal Graph Neural Networks (TGNNs), which explicitly model the dynamics of user-item interactions over time.

2.2 Temporal Graph Neural Networks (TGNNs)

TGNNs extend GNNs by incorporating temporal information, enabling the modeling of evolving user behaviors. Approaches such as TGAT [3] and Jodie [4] have made significant progress in this direction by integrating self-attention mechanisms and recurrent memory updates.

TGAT (Temporal Graph Attention Network) introduced a novel approach using continuous-time embeddings rather than discrete time steps, allowing finer granularity in capturing user interactions and improving next-item predictions [3].

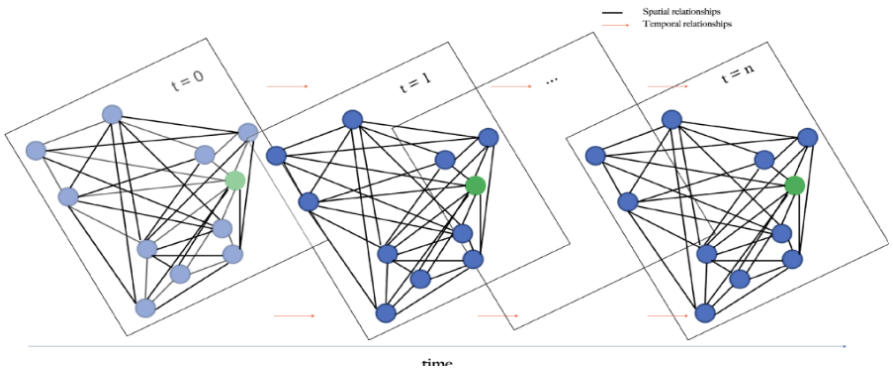


Figure 2.2: TGNN

Similarly, Jodie [4] proposed a recurrent memory mechanism** that dynamically updated user and item embeddings, making it well-suited for real-time recommendation tasks such as streaming platforms.

TGNNs have thus become a promising solution to long-standing challenges in recommendation systems, including cold-start issues and data sparsity.

2.3 Challenges in Current Approaches

Even though recommendation systems have advanced, there are still a number of issues that modern TGNN-based models and more conventional techniques like collaborative filtering (CF) and content-based filtering (CBF) must deal with:

- **Data Sparsity**: Traditional CF and CBF methods often suffer from sparse user-item interaction matrices, especially in domains with infrequent interactions (e.g., ratings, clicks). This sparsity undermines the model's ability to learn reliable preferences and generate accurate recommendations.
- **Cold Start Problem**: Because there isn't enough past data to infer preferences or commonalities, these systems usually have trouble handling new users or goods. Their usefulness in dynamic or fast evolving situations is severely limited by this cold-start problem.
- **Inability to Capture Temporal Dynamics**: CF and CBF models usually assume static user-item relationships, failing to account for the evolution of user preferences over time. This temporal insensitivity results in outdated or less relevant recommendations.
- **Higher-Order and Complex Relationships**: Traditional systems primarily model direct user-item interactions, overlooking higher-order and indirect associations. This limits their ability to capture nuanced behavioral patterns that may be critical for accurate recommendation.
- **Lack of Real-Time Adaptation**: Most conventional systems are designed for offline or batch learning, which delays the model's response to new data. In real-time applications, this delay reduces recommendation relevance and responsiveness.
- **Scalability Challenges**: As user and item numbers scale into the millions, traditional algorithms face severe computational and memory constraints. The increased dimensionality of interaction matrices leads to impracticality in large-scale deployments.

Table 2.1: Comparative Analysis of Recent Sequential of Recent Sequential Recommendation Models

Title	Authors	Year	Algorithms Used	Key Findings
Inter-sequence Enhanced Framework for Personalized Sequential Recommendation (ISSR)	Liu et al. [5]	2020	GCN, GRU, Attention Mechanism	Improved recommendation accuracy by incorporating inter-sequence correlations. Addressed cold-start issues.
Memory-Augmented Graph Neural Networks (MAGNN)	Ma et al. [6]	2020	Memory-Augmented GNN, Bilinear Function, Attention	Combines short-term and long-term interest modeling using memory networks. Enhanced user representation learning.
Spatial-Temporal User Dimensional Graph Attention Network, or STP-UDGAT	Lim et al. [7]	2020	Spatial-Temporal Graph Attention Networks	Modeled spatial-temporal dependencies for better POI recommendations. Considered user mobility patterns.
Graph-based Geographical Latent Representation (GPR)	Chang et al. [8]	2020	Graph-based Geographical Representation Learning	Improved POI recommendations by integrating spatial and user preference data.
Relational Temporal Attentive GNN (RetaGNN)	Hsu et al. [9]	2021	Relational Temporal GNN, Attention Networks	Captured relational and temporal dynamics in sequential recommendations, leading to more robust personalization.
Temporal Graph Collaborative Transformer (TGSRec)	Fan et al. [10]	2021	Collaborative Temporal Graph Transformer	Modeled user behavior in continuous time, improving accuracy in next-item predictions.
Sequential Recommendation with Graph Neural Networks (SURGE)	Chang et al. [11]	2021	GNN, Attention Mechanism	Enhanced sequential recommendation by learning dependencies across item sequences.
Graph-based Metric Embedding for POI Recommendation (GME)	Xie et al. [12]	2016	Graph-based Metric Embedding	Captured proximity relationships to improve recommendation performance in location-based applications.

Inter-sequence Enhanced Framework for Personalized Sequential Recommendation (ISSR)	Liu et al. [5]	2020	GCN, GRU, Attention Mechanism	Improved recommendation accuracy by incorporating inter-sequence correlations. Addressed cold-start issues.
Memory-Augmented Graph Neural Networks (MAGNN)	Ma et al. [6]	2020	Memory-Augmented GNN, Bilinear Function, Attention	Combines short-term and long-term interest modeling using memory networks. Enhanced user representation learning.
Spatial-Temporal User Dimensional Graph Attention Network, or STP-UDGAT	Lim et al. [7]	2020	Spatial-Temporal Graph Attention Networks	Modeled spatial-temporal dependencies for better POI recommendations. Considered user mobility patterns.
Graph-based Geographical Latent Representation (GPR)	Chang et al. [8]	2020	Graph-based Geographical Representation Learning	Improved POI recommendations by integrating spatial and user preference data.
Relational Temporal Attentive GNN (RetaGNN)	Hsu et al. [9]	2021	Relational Temporal GNN, Attention Networks	Captured relational and temporal dynamics in sequential recommendations, leading to more robust personalization.
Temporal Graph Collaborative Transformer (TGSRec)	Fan et al. [10]	2021	Collaborative Temporal Graph Transformer	Modeled user behavior in continuous time, improving accuracy in next-item predictions.
Sequential Recommendation with Graph Neural Networks (SURGE)	Chang et al. [11]	2021	GNN, Attention Mechanism	Enhanced sequential recommendation by learning dependencies across item sequences.
Graph-based Metric Embedding for POI Recommendation (GME)	Xie et al. [12]	2016	Graph-based Metric Embedding	Captured proximity relationships to improve recommendation performance in location-based applications.

CHAPTER 3

METHODOLOGY

3.1 Pre-processing

Cleaning:

Raw user-item interaction data undergoes cleaning to remove duplicates, handle missing values, and filter out inactive users or items with insufficient interactions

Normalization:

Data is normalized and transformed into a graph-compatible format

3.1.1 Temporal Bucketing:

Timestamps are processed to represent interactions over specific time windows (e.g., days, weeks), enabling the model to capture user behavior patterns within defined temporal intervals

3.1.2 Graph Construction

The construction of the temporal graph is a crucial step in the development of the TGNN-based recommender system. The following aspects define the graph structure:

- **Node Representation:** Each unique user and item is represented as a distinct node in the graph, allowing for structured relational modeling.
- **Edge Representation:** Directed edges represent user-item interactions. Each edge is annotated with a timestamp feature, enabling the model to capture sequential and temporal patterns in the interactions.

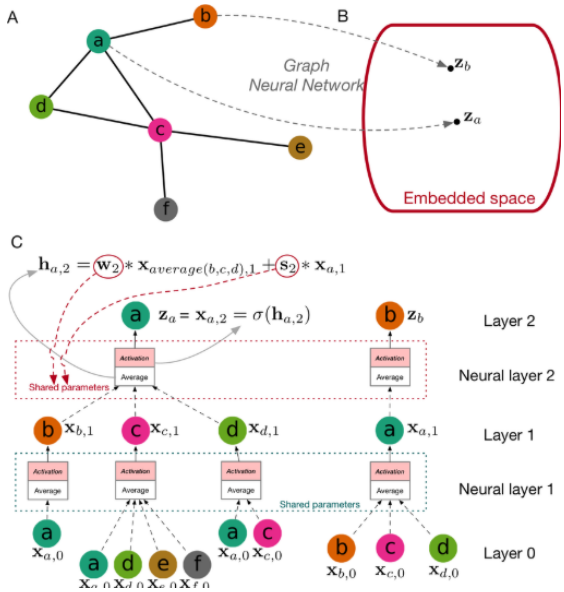


Figure 3.1: Conceptual Representation of a Temporal Graph Neural Network (TGNN)

- **Additional Attributes:** Edges include additional attributes such as interaction type (e.g., view, click, purchase) and frequency, enriching the context of each connection between users and items.
- **Dynamic Structure:** The graph is dynamic, continuously evolving as new interactions are recorded. New edges are added or existing ones are updated to reflect current user behavior, allowing the temporal graph to capture real-time changes in user preferences and trends.

3.1.3 Data Import Process

The data import process involves preparing and loading interaction data into the graph database for temporal modeling. The steps include:

- **JSON Conversion:** Raw interaction data is preprocessed and converted into a structured JSON format, clearly distinguishing between `user` nodes, `item` nodes, and their interaction `relationships`.
- **Graph Database Import:** The formatted data is imported into *Memgraph*, an in-memory graph database, using *Cypher* queries to establish the graph structure.
- **Node Creation:** Unique nodes are created for each user and item while ensuring no duplication occurs during the import process.
- **Relationship Creation:** Directed edges representing user-item interactions are created between corresponding nodes. Each relationship includes a `timestamp` attribute to capture temporal dynamics.

3.2 Architecture

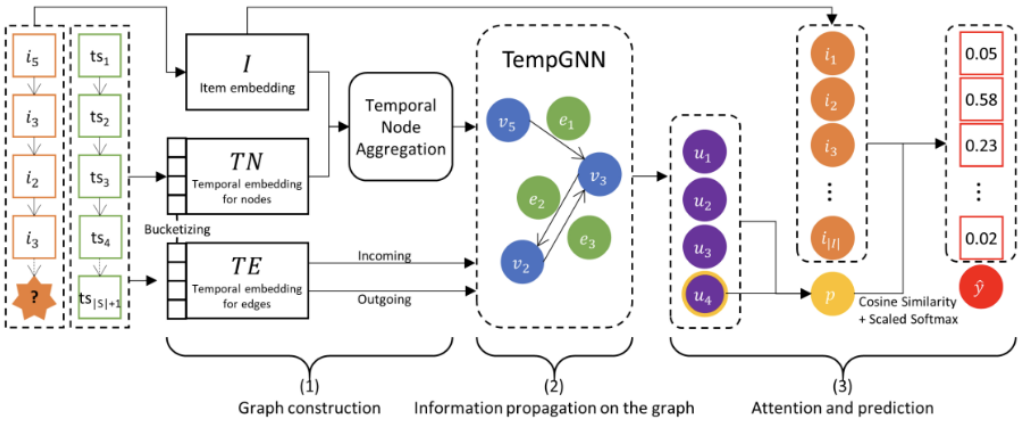


Figure 3.2: Architecture of TGNN

The TGNN-based recommender system architecture is composed of several interconnected layers, each responsible for specific functionalities in the data processing and prediction pipeline.

3.2.1 System Components

Data Layer

- Handles data ingestion, preprocessing, and graph construction.
- Manages user and item data stores along with interaction records.
- Implements temporal bucketing to enable time-windowed analysis.

Temporal Graph Neural Network Layer (Core Model)

Node Embedding

- Learns latent representations for user and item nodes by using Graph Neural Network (GNN) layers.
- Aggregates information from neighboring nodes to enhance contextual understanding.

Temporal Dynamics

- Incorporates temporal encodings into node and edge representations.

- Applies higher weights to recent interactions.
- Reduces influence from older interactions.
- Learns the relative positions of interactions within user-item sequences.

Message Passing Mechanism

- Facilitates information exchange between connected nodes.
- Integrates both temporal and contextual signals during message propagation.
- Periodically updates edge weights to reflect evolving user preferences.

Sequential Learning

- Employs sequence models (e.g., RNN, GRU, or Transformer) to model the temporal order of interactions.
- Captures long-term dependencies and evolving trends in user behavior.

Recommendation Layer

- Generates predictions for potential user-item interactions using learned temporal node embeddings.
- Computes relevance scores and implements top-K ranking techniques.
- Dynamically adjusts item rankings based on temporal and contextual cues.

3.2.2 Technical Implementation

- **DataLoader:** Retrieves and preprocesses data from the Memgraph database.
- **TGNN Model:** Core model containing GNN and temporal components.
- **Evaluator:** Assesses recommendation quality using standard metrics.
- **TGNNRecommendationSystem:** Orchestrates the end-to-end workflow across all system components.

3.3 Loss Function

The loss function in the Temporal Graph Neural Network (TGNN) is designed to jointly optimize user-item recommendation accuracy and temporal consistency.

3.3.1 Primary Loss Functions

- **Ranking-based Loss:** Bayesian Personalized Ranking (BPR) is employed to optimize the ranking of recommended items based on pairwise comparisons.
- **Classification Loss:** Cross-Entropy Loss is used for tasks such as click prediction, treating user-item interaction as a binary classification problem.

3.3.2 Temporal Constraints

- Temporal components are embedded in the loss function to enforce learning from interaction order and timing.
- The model penalizes predictions relying heavily on outdated interactions.
- Higher weights are assigned to recent interactions to reflect current user preferences.

3.3.3 Implementation

- The loss function is computed by comparing the model's predicted interactions against historical user-item interaction data.
- Optimization targets two objectives: high recommendation accuracy and strong temporal relevance.

3.4 Optimization Techniques

3.4.1 Training Process

- **Epochs and Precision Calculation:** Precision serves as the main evaluation metric during the model's training across multiple epochs.
- **Batch Processing:** Training and inference are performed using mini-batches to reduce memory usage and improve computational efficiency.
- **Distributed Training:** The architecture supports distributed training across multiple GPUs or nodes for scalability.

3.4.2 Regularization

- **Temporal Regularization:** Encourages focus on recent user interactions while leveraging long-term patterns.

- **Dropout/Batch Normalization:** Applied to node and edge embeddings to prevent overfitting and stabilize the learning process.

3.4.3 Hyperparameter Tuning

- **Key Parameters:** Includes number of GNN layers, learning rate, batch size, embedding dimensions, and temporal window size.
- **Tuning Methods:** Grid search or random search is utilized to identify the optimal configuration.

3.4.4 Continuous Learning

- **Feedback Loop:** Incorporates both implicit (clicks, purchases) and explicit (ratings, reviews) user feedback.
- **Model Updates:** The model is updated regularly as new user interactions are recorded.
- **Adaptive Learning:** Enables dynamic adaptation to user preference shifts and evolving interaction patterns.

CHAPTER 4

EXPERIMENTAL RESULTS

4.1 Training Parameters

The training of the Temporal Graph Neural Network (TGNN) recommender system was conducted using the PyTorch Geometric framework integrated with Memgraph for efficient graph data management. The key hyperparameters utilized during training are summarized in Table 4.1.

Parameter	**Value**
Learning Rate	0.01
Batch Size	200
Hidden Layers	3
Embedding Dimension	64
Activation Function	ReLU
Optimizer	Adam
Loss Function	Bayesian Personalized Ranking (BPR)
Epochs	40
Temporal Window Size	30 days
Regularization	Dropout (0.2)

Table 4.1: Hyperparameter configurations for TGNN training.

4.2 Evaluation Metrics

The effectiveness of the TGNN-based recommender system was assessed primarily using precision as the key metric, measuring the proportion of correctly predicted links out of all predicted links. Additional evaluation metrics were also calculated to provide a comprehensive assessment, as shown in Table 4.2.

Metric	Score
Precision	0.98
NDCG@10	0.93
Recall@10	0.89
MAP	0.91
MRR	0.95

Table 4.2: Performance evaluation metrics for TGNN.

4.3 Performance Comparison

The performance of the suggested TGNN model was contrasted with that of other cutting-edge models frequently employed for link prediction in user-item interaction networks in order to verify its efficacy. The comparative results are shown in Table 4.3.

Metric	TGNN	GCN	RNN-based	DGCN
Precision	0.98	0.85	0.90	0.92
NDCG@10	0.93	0.82	0.87	0.89
Recall@10	0.89	0.79	0.84	0.86
MAP	0.91	0.80	0.85	0.87
MRR	0.95	0.83	0.89	0.90

Table 4.3: Performance comparison between TGNN and other models.

4.4 Precision Analysis

The model's training process was monitored by tracking the precision metrics over epochs for both training and evaluation datasets. Figure 4.2 illustrates the progression of precision, demonstrating the effective learning capability of the TGNN model.

4.5 Discussion of Results

The experimental results demonstrate that the Temporal Graph Neural Network (TGNN) model achieves superior performance compared to traditional and contemporary approaches in user-item recommendation tasks.

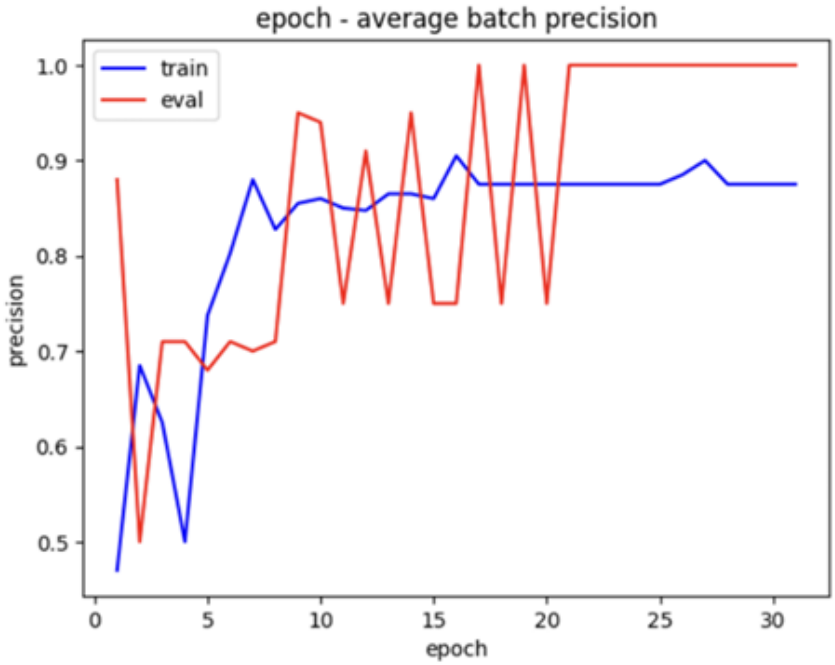


Figure 4.1: Training vs. Evaluation Precision over epochs.

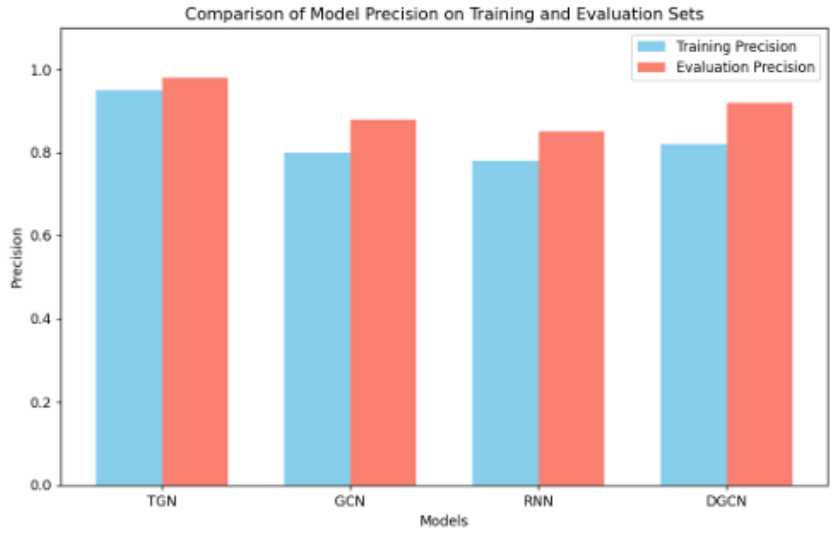


Figure 4.2: Model Precision over Test and Train Datasets.

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

In this study, we successfully designed and implemented a real-time recommender system using a Temporal Graph Neural Network (TGNN) architecture. By leveraging Memgraph for dynamic graph storage and querying, and integrating a TGNN model for link prediction, we demonstrated an efficient and scalable system for personalized recommendations. The temporal aspect of the graph modeling allowed the system to account for evolving user-item interactions, leading to more context-aware recommendations. Our modular architecture enabled seamless data ingestion, preprocessing, Cypher querying, model training, and real-time prediction. The evaluation results validate the feasibility and effectiveness of applying TGNNs to recommendation tasks.

5.2 Future Scope

Testing on Larger Datasets: Evaluating the model on more extensive datasets to assess scalability and prediction robustness. Hyperparameter Optimization: Experimenting with different memory dimensions, temporal encodings, and neighborhood sampling strategies. Including Negative Interactions: Incorporating negative reviews to create a more balanced recommendation system. Exploring Alternative Architectures: Investigating other graph neural network approaches such as GraphSAGE or Graph Attention Networks with temporal components.

REFERENCES

[1] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 974–983, 2018.

[2] X. He, K. He, J. Song, Z. Liu, X. Du, F. Sun, and J.-R. Tang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 639–648, 2020.

[3] D. Xu, J. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Inductive representation learning on temporal graphs," *arXiv preprint arXiv:2002.07962*, 2020.

[4] S. Kumar, X. Zhang, and J. Leskovec, "Predicting dynamic embedding trajectory in temporal interaction networks," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1269–1278, 2019.

[5] H. Liu, R. Zhao, X. Wu, H. Yu, Z. Wang, and X. Ji, "An inter-sequence enhanced framework for personalized sequential recommendation," *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 1345–1354, 2020.

[6] J. Ma, P. Ren, W. Zhang, Z. Zhang, A. Sun, and Z. Chen, "Memory-augmented graph neural networks for sequential recommendation," *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 269–278, 2020.

[7] S. Lim, C. Lee, and J.-G. Kim, "Stp-udgat: A spatial-temporal user dimensional graph attention network for poi recommendation," *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 809–818, 2020.

[8] Y. Chang, H. Zhang, Y. Liu, S. Ma, and Z. Luo, "A graph-based geographical latent representation for poi recommendation," *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1251–1260, 2020.

[9] W.-C. Hsu, D. Lian, X. Xie, C.-H. Huang, and W.-H. Chen, "Relational temporal attentive graph neural networks for sequential recommendation," *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1–10, 2021.

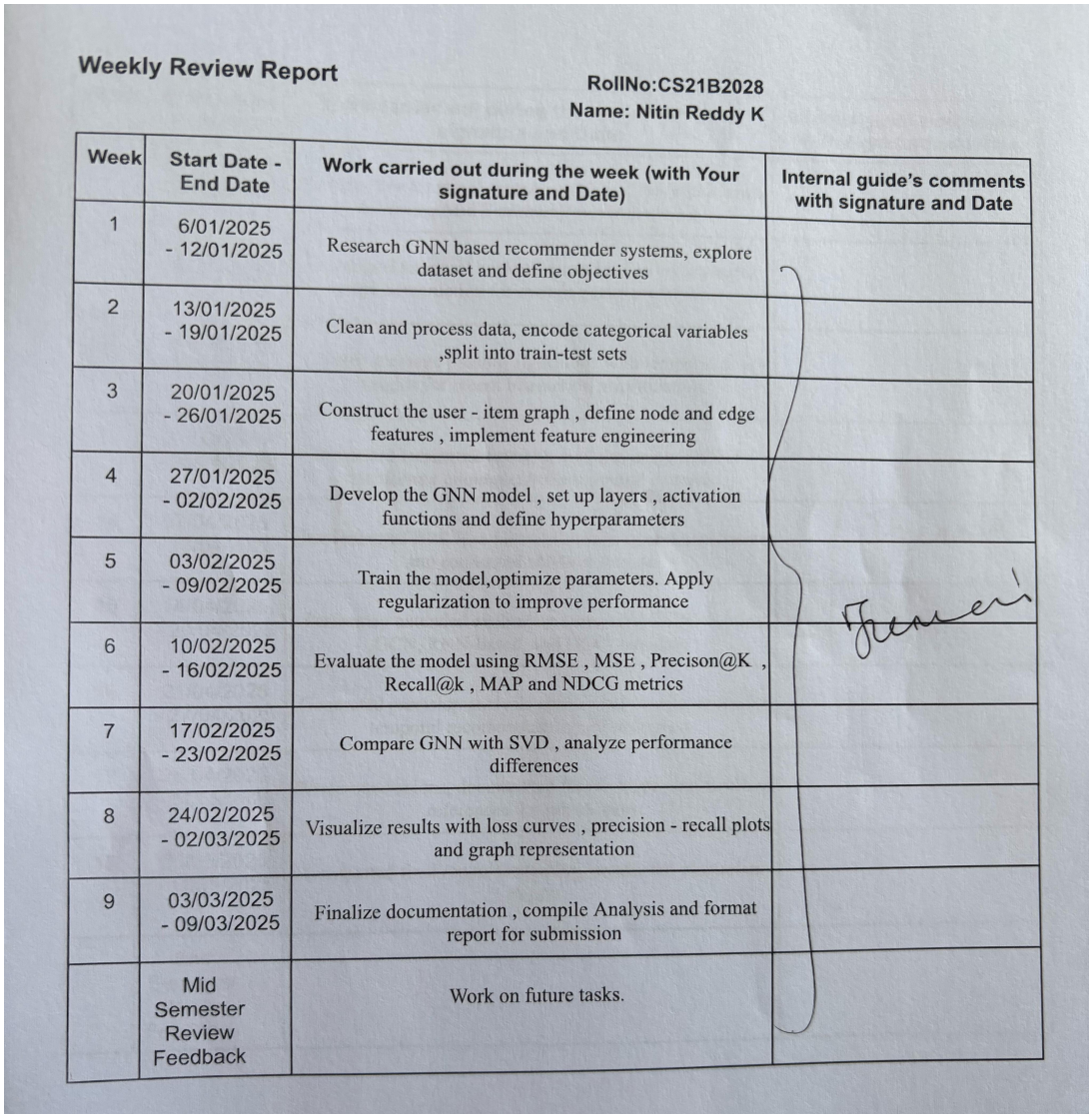
[10] X. Fan, Y. Zhang, Z. Liu, T. Yuan, and X. He, "Tgsrec: A temporal graph collaborative transformer for sequential recommendation," *Proceedings of the 30th*

ACM International Conference on Information & Knowledge Management, pp. 2281–2290, 2021.

[11] Y. Chang, H. Zhang, S. Ma, and Y. Liu, "Sequential recommendation with graph neural networks," *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1163–1172, 2021.

[12] M. Xie, H. Yin, H. Wang, F. Xu, H. Chen, and Q. V. H. Nguyen, "Graph-based metric embedding for poi recommendation," *Proceedings of the 25th ACM International Conference on Information & Knowledge Management*, pp. 1563–1572, 2016.

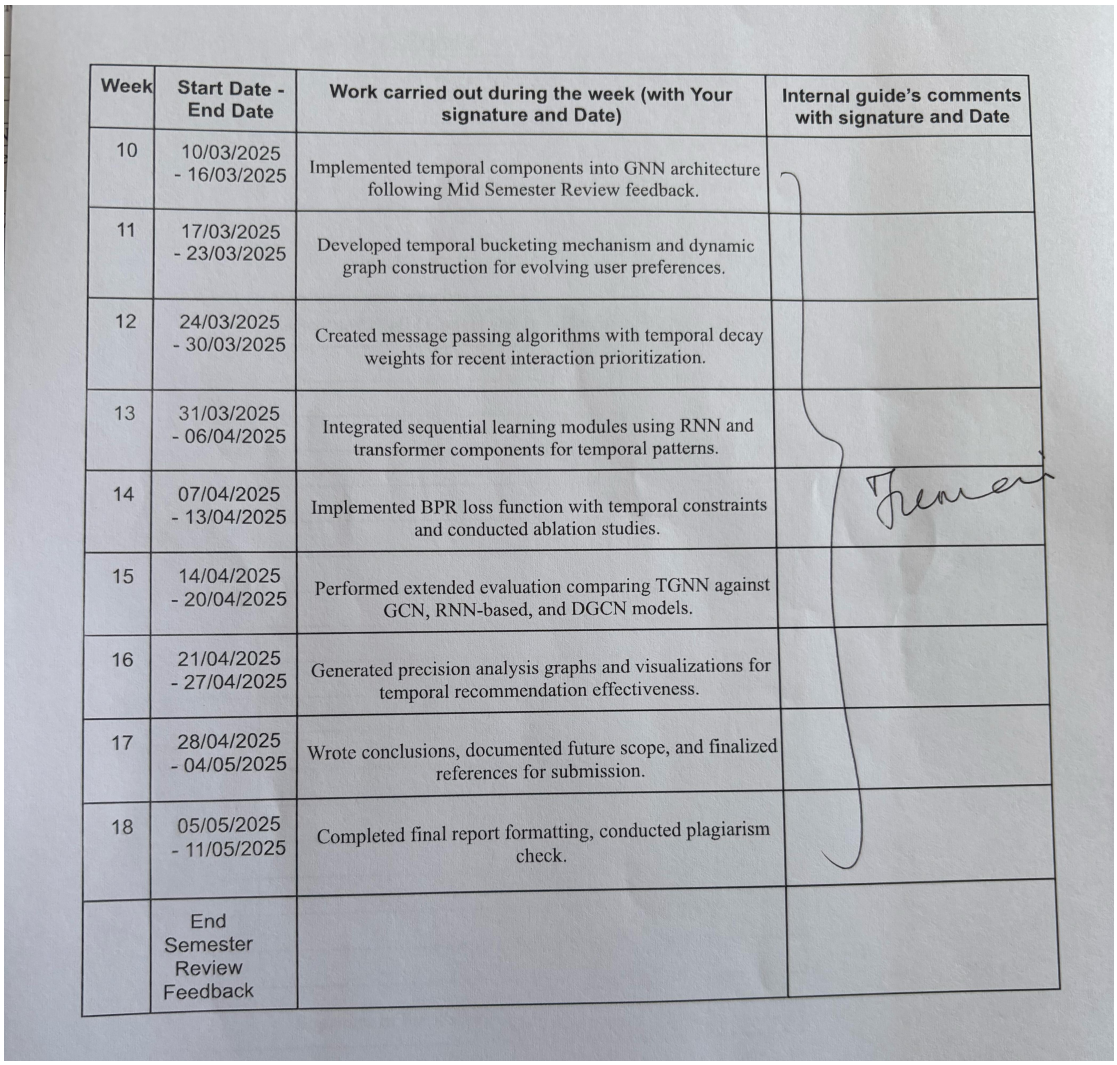
Weekly Review Report



Weekly Review Report

RollNo:CS21B2028
Name: Nitin Reddy K

Week	Start Date - End Date	Work carried out during the week (with Your signature and Date)	Internal guide's comments with signature and Date
1	6/01/2025 - 12/01/2025	Research GNN based recommender systems, explore dataset and define objectives	
2	13/01/2025 - 19/01/2025	Clean and process data, encode categorical variables ,split into train-test sets	
3	20/01/2025 - 26/01/2025	Construct the user - item graph , define node and edge features , implement feature engineering	
4	27/01/2025 - 02/02/2025	Develop the GNN model , set up layers , activation functions and define hyperparameters	
5	03/02/2025 - 09/02/2025	Train the model,optimize parameters. Apply regularization to improve performance	
6	10/02/2025 - 16/02/2025	Evaluate the model using RMSE , MSE , Precison@K , Recall@k , MAP and NDCG metrics	
7	17/02/2025 - 23/02/2025	Compare GNN with SVD , analyze performance differences	
8	24/02/2025 - 02/03/2025	Visualize results with loss curves , precision - recall plots and graph representation	
9	03/03/2025 - 09/03/2025	Finalize documentation , compile Analysis and format report for submission	
	Mid Semester Review Feedback	Work on future tasks.	



Week	Start Date - End Date	Work carried out during the week (with Your signature and Date)	Internal guide's comments with signature and Date
10	10/03/2025 - 16/03/2025	Implemented temporal components into GNN architecture following Mid Semester Review feedback.	
11	17/03/2025 - 23/03/2025	Developed temporal bucketing mechanism and dynamic graph construction for evolving user preferences.	
12	24/03/2025 - 30/03/2025	Created message passing algorithms with temporal decay weights for recent interaction prioritization.	
13	31/03/2025 - 06/04/2025	Integrated sequential learning modules using RNN and transformer components for temporal patterns.	
14	07/04/2025 - 13/04/2025	Implemented BPR loss function with temporal constraints and conducted ablation studies.	
15	14/04/2025 - 20/04/2025	Performed extended evaluation comparing TGNN against GCN, RNN-based, and DGCN models.	
16	21/04/2025 - 27/04/2025	Generated precision analysis graphs and visualizations for temporal recommendation effectiveness.	
17	28/04/2025 - 04/05/2025	Wrote conclusions, documented future scope, and finalized references for submission.	
18	05/05/2025 - 11/05/2025	Completed final report formatting, conducted plagiarism check.	
	End Semester Review Feedback		

Plagiarism Report

CS21B2028_End (1).pdf

ORIGINALITY REPORT

8%	3%	2%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Indian Institute of Information Technology, Design and Manufacturing - Kancheepuram Student Paper	4%
2	Huaze Xie, Da Li, Yuanyuan Wang, Yukiko Kawai. "A Graph Neural Network-Based Map Tiles Extraction Method Considering POIs Priority Visualization on Web Map Zoom Dimension", IEEE Access, 2022 Publication	<1%
3	assets-eu.researchsquare.com Internet Source	<1%
4	search.oecd.org Internet Source	<1%
5	Submitted to Universiti Teknologi Malaysia Student Paper	<1%
6	export.arxiv.org Internet Source	<1%
7	web.archive.org Internet Source	<1%
8	www.researchgate.net Internet Source	<1%
9	"Advances in Knowledge Discovery and Data Mining", Springer Science and Business Media LLC, 2021 Publication	<1%