EXPLORING MULTI-VIEW GRAPH NEURAL NETWORK (MV-GNN) FOR GRAPH COLORING

By
MUSFIQUR RAHMAN SAHIDUL
ID: 201-15-13832

MD. MINHAJUL ISLAM
ID: 201-15-14061

FINAL YEAR DESIGN PROJECT REPORT

This Report Presented in Partial Fulfillment of the Requirements for the **Degree of Bachelor of Science in Computer Science and Engineering**

Supervised by
Md. Ferdouse Ahmed Foysal
Lecturer
Department of Computer Science and Engineering Daffodil International University

Co-Supervised by

Dewan Mamun Raza
Assistant Professor
Department of Computer Science and Engineering Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY
Dhaka, Bangladesh

January 12, 2025

APPROVAL

This Project titled "**EXPLORING MULTI-VIEW GRAPH NEURAL NETWORK (MV-GNN) FOR GRAPH COLORING**" submitted by Mushfiqur Rahman Sahidul, ID No: **201-15-13832** and Md. Minhajul Islam, ID No: **201-15-14061** to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on **12 January, 2025**.

BOARD OF EXAMINERS

Dr. Md. Fokhray Hossain — **Chairman**
Professor
Department of CSE, FIST
Daffodil International University

Mr. Shah Md. Tanvir Siddiquee — **Internal Examiner**
Assistant Professor
Department of CSE, FSIT
Daffodil International University

Mr. Md Umaid Hasan — **Internal Examiner**
Lecturer (Senior Scale)
Department of CSE, FSIT
Daffodil International University

Nazibur Rahman — **External Examiner**
Technical Lead - Database Administrator
Telenor - Grameen Phone Account

DECLARATION

We hereby declare that this project has been done by us under the supervision of **Md. Ferdouse Ahmed Foysal , Lecturer**, Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Supervised by:

11/01/25

Md. Ferdouse Ahmed Foysal
Lecturer
Department of Computer Science and
Engineering Daffodil International University

Co-Supervised by:

12.01.2025

Dewan Mamun Raza
Assistant Professor
Department of Computer Science and
Engineering Daffodil International University

Submitted by:

Mushfiqur 12-01-2025

Mushfiqur Rahman Sahidul
Student ID: 201-15-13832
Department of Computer Science and
Engineering Daffodil International University

Minhaj 12-01-2025

Md. Minhajul Islam
Student ID: 201-15-14061
Department of Computer Science and
Engineering Daffodil University

ACKNOWLEDGEMENTS

This work would not have been possible without the support and contributions of many individuals over the past two semesters. We are deeply grateful to everyone who has assisted us in one way or another.

First, we express our heartfelt thanks and gratefulness to the almighty for His divine blessing making it possible for us to complete the **Final Year Design Project (FYDP)** successfully.

We are grateful and wish our profound indebtedness to **Md. Ferdouse Ahmed Foysal**, **Lecturer**, Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh. Deep knowledge and keen interest of our supervisor in the field of **"Graph Neural Networks"** to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

We would like to express our heartfelt gratitude to the Head of the Department of Computer Science and Engineering, for his kind help in finishing our project and also to other faculty members and the staff of the Department of Computer Science and Engineering, Daffodil International University.

We would like to thank our entire course-mates at Daffodil International University, who took part in this discussion while completing the coursework.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

ABSTRACT

The Graph Coloring Problem (GCP), which involves determining the chromatic number or the minimum number of colors needed to color neighboring nodes in a graph, is a critical computational challenge with applications in scheduling, resource allocation, and frequency assignment. Heuristic-based algorithms like Artificial Bee Colony (ABC), Sequential Coloring Algorithm (SCA), Welsh–Powell Algorithm (WPA), Branch-and-Cut, Greedy, Dsatur, RLF, and others are commonly used to tackle this problem, but they are computationally expensive and often do not yield optimal solutions. Given the graph-structured nature of the problem, graph-based models such as Graph Neural Networks (GNNs) have proven to be highly advantageous. In this research, we extend GNNs to Multi-View Graph Neural Networks (MV-GNNs) to address the GCP. By leveraging multiple perspectives of a graph, our approach incorporates distinct views to learn node embeddings and predict optimal color assignments. Through extensive experiments and comparisons with existing state-of-the-art algorithms, we demonstrate that MV-GNNs can efficiently determine the chromatic number of large graphs, achieving competitive performance in coloring accuracy, scalability, and computational efficiency. This work represents a foundational step in applying multi-view learning paradigms to solve NP-complete problems.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

The Graph Coloring Problem (GCP) is one of the most fundamental and challenging problems in graph theory, widely regarded as an NP-Complete problem. Its complexity arises from the requirement to assign the minimum number of colors to the nodes of a graph such that no two adjacent nodes share the same color. This task, referred to as determining the chromatic number, is computationally intensive and has applications across a diverse range of domains, including air traffic control, biology, computer science, physics, chemistry, psychology, autonomous route finding, urban planning, protein-protein interaction networks, and scheduling.[1]

Finding the optimal solution for GCP in polynomial time is not feasible due to its NP-hard nature. Consequently, researchers have developed various heuristic and metaheuristic algorithms to approximate solutions efficiently. Common heuristic approaches include the Welsh-Powell algorithm, which orders vertices by degrees; the DSatur algorithm, which dynamically assigns colors based on saturation degrees; and greedy strategies like the First Fit algorithm. Similarly, metaheuristic techniques such as Simulated Annealing (SA), Genetic Algorithms (GA), Ant Colony Optimization (ACO), Tabu Search (TS), and Cuckoo Optimization Algorithm (COA) have been extensively explored. While these methods offer significant improvements in computational efficiency, they often lack adaptability and fail to guarantee optimal solutions, particularly for large-scale and complex graphs. The limitations of traditional algorithms have motivated the adoption of deep learning techniques, particularly Graph Neural Networks (GNNs), to tackle GCP. Unlike classical methods, GNNs leverage graph structures to learn relational patterns and dependencies, providing a robust framework for optimization tasks. However, most existing GNN-based approaches are constrained by their reliance on single-view representations, which may fail to capture the diverse structural and contextual features of complex graphs. To address these limitations, this research proposes the use of Multi-View Graph Neural Networks (MV-GNNs) to solve the graph coloring problem [2]. By incorporating multiple views of the graph, such as adjacency-based, feature-based, and structural representations, MV-GNNs enhance the expressiveness and scalability of GNNs. This approach enables the model to learn from diverse perspectives, improving its ability to generalize and perform well across various graph structures. The motivation for adopting MV-GNNs lies in their potential to significantly improve coloring accuracy, reduce conflicts, and achieve computational efficiency, thereby bridging the gap between theoretical advancements and practical applications.

The introduction of MV-GNNs to the graph coloring domain represents a significant step

forward in addressing the challenges associated with NP-complete problems. This research not only contributes to the development of efficient algorithms for GCP but also opens new avenues for applying multi-view learning paradigms to other optimization problems in real-world contexts. By leveraging the power of deep learning and graph-based representations, this study aims to advance the state-of-the-art in graph coloring and establish a foundation for future research in the field.

1.2 Motivation

The motivation for employing Multi-View Graph Neural Networks (MV-GNNs) to solve the Graph Coloring Problem (GCP) arises from the inherent challenges posed by traditional methods and the promising capabilities of deep learning in addressing these challenges.

Classical approaches to solving GCP, such as greedy algorithms, backtracking, and metaheuristic techniques (e.g., Genetic Algorithms, Tabu Search, and Ant Colony Optimization), suffer from several limitations. While these methods are useful for small or moderately complex graphs, they often exhibit scalability issues, high computational overhead, and reduced adaptability when applied to large, complex, or dynamically changing graph structures. Additionally, such methods rely heavily on handcrafted heuristics and fixed rules, which limit their ability to generalize across different types of graph data. On the other hand, Graph Neural Networks (GNNs) have demonstrated remarkable potential in capturing complex dependencies and relational information in graph-structured data. GNNs offer a data-driven approach to learning graph representations, enabling them to adapt to diverse graph structures without relying on predefined heuristics. By directly learning patterns from data, GNNs address the limitations of traditional methods and offer scalability and flexibility for solving combinatorial optimization problems.

However, single-view GNNs, while effective, can still fall short in capturing the full complexity of graph structures. Graphs often contain multiple layers of information—structural, topological, and feature-based—that a single view may fail to represent comprehensively. This gap highlights the need for a more robust and expressive approach, which brings us to the concept of multi-view learning.

The use of MV-GNNs is motivated by their ability to incorporate multiple perspectives of a graph, such as adjacency-based views, feature-based views, and structural representations. Each view captures unique aspects of the graph, and combining these views allows the model to learn richer and more nuanced node embeddings. The attention mechanism used in MV-GNNs further enhances this capability by dynamically weighting the importance of each view, ensuring the model focuses on the most relevant features for a given task. By addressing the limitations of traditional methods and leveraging the strengths of MV-GNNs, this research aims to develop a scalable, efficient, and adaptive solution for the GCP. The motivation extends beyond theoretical interest, as advancements in graph coloring have significant implications for real-world applications, including resource scheduling, frequency assignment, register allocation, and optimization in network design. Improved solutions to GCP can lead to more efficient resource

utilization, cost savings, and enhanced performance across various domains. In essence, the motivation for this work stems from the desire to bridge the gap between traditional graph coloring techniques and modern deep learning paradigms, offering a novel, data-driven solution that not only addresses the current challenges but also paves the way for future advancements in graph-based optimization problems.

1.3 Objectives

1. Development of an MV-GNN-based Graph Coloring Framework:

- Design and implement a computational model leveraging MV-GNN to solve the Graph Coloring Problem (GCP).
- Utilize graph datasets sourced from the Stanford Network Analysis Project (SNAP) repository, covering diverse real-world networks such as social, collaboration, and communication graphs.
- Incorporate adjacency-based graph representations for feature extraction and effective message passing to identify optimal vertex color assignments.
- Ensure the framework can handle varying graph sizes and complexities to demonstrate its robustness and versatility.

2. Empirical Evaluation and Performance Analysis:

- Conduct rigorous evaluations of the MV-GNN framework using datasets collected from SNAP, including real-world and large-scale graph structures.
- Benchmark the performance of the MV-GNN model against traditional graph coloring techniques, such as Greedy Coloring and DSatur, in terms of coloring accuracy, conflict resolution, and runtime efficiency.
- Analyze the model's generalization ability across different graph types and assess its scalability for complex network topologies.

3. Visualization and Insights:

- Leverage visualization tools like NetworkX to depict graph structures and coloring outcomes, showcasing conflict minimization and color distribution.
- Generate visual analytics to represent node embeddings, coloring assignments, and clustering patterns to illustrate the model's interpretability and effectiveness.

4. Performance Insights and Model Contributions:

- Investigate the impact of graph views and attention-based message aggregation on the overall graph coloring outcomes.
- Conduct an ablation study to evaluate the contribution of adjacency representations to model performance and conflict resolution efficiency.
- Provide a comprehensive understanding of how multi-view representation techniques contribute to optimization, enhancing solutions for NP-complete problems like GCP

1.4 Methodology

This research introduces a novel approach to solving the Graph Coloring Problem (GCP) using Multi-View Graph Neural Networks (MV-GNNs). The methodology encompasses a systematic and structured framework that integrates multiple perspectives of graph data, advanced graph neural network architectures, and optimization techniques. Below, the methodological components are outlined in detail.

1.4.1 Problem Definition

The Graph Coloring Problem is a combinatorial optimization problem that involves assigning colors to nodes of a graph such that no two adjacent nodes share the same color. The primary goal is to minimize the total number of colors required, referred to as the chromatic number, while ensuring minimal conflicts between adjacent nodes. This problem is NP-complete, making it computationally intractable for large graphs using traditional methods. Applications include resource allocation, scheduling, and frequency assignment, underscoring the practical significance of solving GCP efficiently.

Objective: Develop a robust MV-GNN model capable of learning graph patterns from multi-view representations to achieve efficient and accurate graph coloring solutions across synthetic and real-world datasets.

1.4.2 Data Preparation

Data for this research was collected from the **Stanford Network Analysis Project (SNAP),** which provides a wide range of real-world graph datasets, including social, collaboration, and citation networks.

Graph Representation: Graphs are represented in adjacency list format, suitable for message-passing operations within GNN models.

Feature Extraction: Node features such as degrees, clustering coefficients, and centrality measures are computed where applicable.

Graph Normalization: Standard preprocessing techniques such as graph sparsification and normalization are applied to ensure data consistency and facilitate efficient training

Graph Views: The multi-view representation includes adjacency-based views (capturing direct connectivity), feature-based views (highlighting node attributes), and structural views (identifying topological patterns like clusters and neighborhoods).

1.4.3 Model Architecture

The proposed model leverages Multi-View Graph Neural Networks (MV-GNNs) to address GCP through attention-based aggregation mechanisms.

- **Graph Views**:
  - Adjacency View: Captures direct connections between nodes.
  - Feature View: Encodes node features like degree and clustering coefficients.
  - Structural View: Reflects higher-order graph properties.
- **View Aggregation**: An attention mechanism aggregates the embeddings from all views into a unified representation. This mechanism dynamically assigns importance to each view based on its relevance, ensuring that the final representation integrates the most informative features.
- **GNN Layers:** The architecture uses multiple layers of message passing networks to propagate and aggregate node information across the graph.
- **Color Prediction**: A fully connected layer processes the aggregated representation to predict color assignments for each node. This layer ensures the model outputs optimal color predictions, minimizing conflicts between adjacent nodes.

1.4.4 Training

Training the MV-GNN model involves optimizing a loss function designed to minimize graph coloring conflicts while maintaining computational efficiency.

Training Details:

- **Loss Function:**
  - A custom loss function penalizes edge conflicts where adjacent nodes are assigned the same color.
  - Regularization terms are included to prevent overfitting and enhance generalization.
- **Optimizer:** Adam optimizer is employed to update model weights efficiently during backpropagation.
- **Training Schedule:** The model is trained over multiple epochs with early stopping criteria to avoid overfitting.
- **Batching:** Mini-batching is used for large datasets to optimize memory usage and computational efficiency.

1.4.5 Evaluation

Evaluation involves assessing the model's performance using both synthetic and real-world datasets. Key metrics for evaluation include:

- **Coloring Accuracy**: The percentage of correctly assigned vertex colors without conflicts.
- **Conflict Rate**: The proportion of adjacent node pairs that share the same color.
- **Runtime Efficiency**: The time required for graph coloring on various graph sizes.
- **Scalability**: The ability of the model to handle large-scale graph datasets.
- **Generalization**: The model's performance across different graph types and unseen datasets

1.4.6 Experimental Setup

The experimental setup is meticulously designed to ensure a systematic and comprehensive evaluation of the proposed Multi-View Graph Neural Network (MV-GNN) model for solving the Graph Coloring Problem (GCP). The key aspects of the experimental setup are outlined below:

GPU-Accelerated Environments:

Experiments were conducted in GPU-accelerated environments utilizing frameworks like PyTorch Geometric. This approach enabled efficient processing of large graph datasets and expedited training and evaluation cycles, essential for graph-based neural models.

Datasets:

Graph datasets used for training and evaluation were sourced from the Stanford Network Analysis Project (SNAP), ensuring a wide range of graph types and complexities:

- **Synthetic Datasets:** Randomly generated graphs using models such as:
    - *Erdős–Rényi Model:* Generates graphs with randomly placed edges, enabling analysis under random connectivity conditions.
    - *Barabási–Albert Model:* Simulates scale-free networks characterized by preferential attachment, suitable for testing on highly connected structures.
- **Real-World Datasets:** Graphs representing social networks, collaboration networks, and other complex real-world systems were used to validate the model's performance under practical conditions.

Validation Approach:

To ensure robust evaluation and mitigate overfitting risks, the following validation strategies were employed:

- **Cross-Validation:**
    - 5-Fold Cross-Validation was adopted to evaluate the model by partitioning each dataset into five segments. The model was iteratively trained on four segments and validated on the fifth, ensuring comprehensive coverage.
- **Baseline Comparisons:**
    - The MV-GNN model's performance was benchmarked against classical graph coloring algorithms (e.g., Greedy Coloring, Welsh–Powell, and DSatur) and single-view GNN models.
    - Metrics such as accuracy, scalability, runtime, and conflict minimization were evaluated to highlight the model's improvements over traditional and comparable approaches.

Ablation Studies:

Ablation studies were conducted to assess the significance of each component within the MV-GNN framework:

- **Component Analysis:**
    - The impact of individual graph views (e.g., adjacency-based, feature-based) was evaluated by selectively removing views during training.
    - The effectiveness of the attention-based view aggregation mechanism was

tested by replacing it with simple averaging techniques.

- **Performance Metrics:** Key performance indicators were monitored to determine the contributions of various views and aggregation strategies

Hyperparameter Tuning:

Optimal performance was achieved through systematic tuning of key hyperparameters:

- **Learning Rate:** Adjusted for stable convergence.
- **Dropout Rate:** Applied to reduce overfitting.
- **Number of Layers:** Configured to balance complexity and performance.
- **Optimizer:** The Adam optimizer was used with weight decay regularization to enhance generalization capabilities.

1.4.7 Visualization

Visualization techniques were employed to interpret the results and gain insights into the MV-GNN model's effectiveness in solving the GCP. These visualizations provided a deeper understanding of node relationships and conflict resolution within the graph.

Graph Coloring Visualizations

- **NetworkX Library:**
  - Graph visualizations were created to depict node coloring before and after model application.
  - Distinct colors assigned to nodes illustrated successful conflict resolution and accurate color assignments by the MV-GNN model.

Embedding Visualizations:

- **t-SNE (t-Distributed Stochastic Neighbor Embedding):**
  - Applied to reduce high-dimensional node embeddings to 2D or 3D visualizations.
  - Clusters of similarly colored nodes indicated effective feature learning and structural pattern recognition.

Attention Weights:

- The attention mechanism's weights were visualized to reveal the contribution of each graph view to the model's final predictions.
- Insights from these visualizations highlighted which graph perspectives were most critical for successful graph coloring.

Performance Trends:

- **Trend Analysis:**
  - Line graphs and bar charts were created to display trends in metrics such as coloring accuracy, conflict rate, and runtime.
  - Comparative visualizations emphasized the model's improvements over traditional algorithms.

These comprehensive visualization techniques validated the model's performance, provided

insights into its operational mechanics, and highlighted potential areas for future improvement in multi-view graph-based learning systems.

1.5 Project Outcome

The successful implementation of the Multi-View Graph Neural Network (MV-GNN) for solving the Graph Coloring Problem (GCP) has demonstrated several key outcomes that highlight its potential as a state-of-the-art solution. The outcomes of this project are summarized below:

Improved Coloring Accuracy

The implementation of the MV-GNN model has significantly enhanced the accuracy of graph coloring by leveraging multi-view perspectives. By combining local constraints, global color distribution, and degree-based features, the model ensures an optimal distribution of colors across nodes while minimizing conflicts.

Conflict Reduction

The MV-GNN architecture incorporates a loss function that penalizes color conflicts between adjacent nodes. Additionally, the multi-view approach ensures a comprehensive understanding of the graph's topology, reducing the number of conflicting edges. In experiments, the model consistently achieves zero or near-zero conflicts for various test graphs.

Scalability

The model demonstrates scalability for graphs of varying sizes, from small graphs to larger, more complex structures. Techniques such as attention mechanisms and pooling layers enhance computational efficiency, enabling the model to process graphs with thousands of nodes effectively.

Ablation Study Insights

The ablation study evaluates the contribution of each view:

- Local Constraint View: Crucial for reducing immediate neighborhood conflicts. Removing this view increases conflicts.
- Global Color Distribution View: Balances color usage across the graph. Ablating this component results in uneven color distributions.
- Degree-Based Constraint View: Ensures high-degree nodes receive distinct colors. Omitting this view reduces the model's performance for graphs with hubs.

Generalization

The MV-GNN model generalizes well across different graph types, including random, scale-free, and planar graphs. Its ability to adapt to diverse topologies demonstrates its robustness and applicability to real-world problems.

Visualization and Interpretability

Graph visualizations generated using the model provide insights into the coloring process. The use of distinct color assignments and layout algorithms highlights the effectiveness of the multi-view approach. Additionally, metrics such as color distribution variance and conflict statistics offer interpretability.

Benchmark Performance

The MV-GNN model has been tested against traditional graph coloring algorithms and baseline machine learning approaches. It outperforms benchmarks in terms of accuracy, conflict reduction, and computational efficiency. Comparisons against algorithms like greedy coloring and genetic algorithms validate its superiority.

Potential Applications

- Scheduling Problems: Assigning time slots in exams or job scheduling without conflicts.
- Frequency Assignment: Minimizing interference in wireless networks.
- Map Coloring: Ensuring no two adjacent regions share the same color.
- Resource Allocation: Optimizing tasks or resources in distributed systems.
- Data Visualization: Enhancing clarity in graphical data representation.

The outcomes of this project not only demonstrate the effectiveness of the MV-GNN model but also establish a strong foundation for future advancements in solving NP-complete problems through multi-view learning.

1.6 Organization of the Report

Chapter 1: Introduction

The introductory chapter provides an overview of the graph coloring problem, emphasizing its importance in combinatorial optimization and various practical applications. It discusses traditional algorithms like greedy and heuristic approaches, outlining their limitations in terms of scalability and accuracy. This chapter motivates the use of graph neural networks and introduces the MV-GNN model as a novel solution. The objectives and scope of the research are clearly defined, setting the stage for the subsequent chapters.

Chapter 2: Background

Graph coloring represents a fundamental problem in graph theory, requiring vertices to be colored such that no adjacent vertices share colors while minimizing the total colors used. Traditional approaches range from simple greedy algorithms to complex backtracking methods, each facing trade-offs between solution quality and computational efficiency. The advent of Graph Neural Networks (GNNs) has introduced new possibilities by enabling learning directly from graph structures through message passing between nodes.

Our Multi-View Graph Neural Network (MV-GNN) approach builds upon these foundations by combining local constraints, global color distribution, and degree-based information to achieve more effective solutions. This synthesis of classical graph theory with modern deep learning techniques provides a promising framework for addressing the limitations of traditional coloring methods.

Chapter 3: Methodology

This chapter delves into the design and components of the MV-GNN model. The local constraint view captures neighborhood-level relationships using GCNConv layers and multihead attention, ensuring minimal conflicts among adjacent nodes. The global color distribution view employs pooling mechanisms to analyze the graph holistically, maintaining balanced color usage across the entire graph. The degree-based constraint view applies weights based on node degree, giving high-degree nodes priority for distinct colors. The chapter also details the multi-component loss function, which integrates penalties for local constraints, global distribution, degree-based conflicts, and excessive color usage, ensuring a comprehensive approach to graph coloring.

Chapter 4: Implementation

The implementation chapter focuses on the technical realization of the MV-GNN model. It begins with node feature engineering, where attributes like normalized degree, clustering coefficient, and eigenvector centrality are computed to enrich the graph representation. The MV-GNN architecture is then detailed, highlighting its GCNConv layers, attention mechanisms, and fusion strategies for combining views. Training methodologies, including the AdamW optimizer, learning rate scheduling, and early stopping, are described, along with the rationale for hyperparameter choices. Finally, the evaluation metrics—accuracy, conflict count, and color usage—are introduced as key performance indicators.

Chapter 5: Results and Analysis

This chapter presents a comprehensive analysis of experimental results. It compares the MV-GNN model's performance with theoretical chromatic bounds and traditional algorithms, showcasing its superior accuracy and conflict minimization. Ablation studies dissect the contributions of each view, quantifying their impact on overall performance. Scalability tests demonstrate the model's efficiency in handling graphs of various sizes, while case studies provide detailed visualizations of the coloring process for specific graph instances. The insights gained underscore the MV-GNN's effectiveness and practical relevance.

Chapter 6: Conclusion and Future Work

The final chapter summarizes the research contributions, emphasizing the MV-GNN's advancements in graph coloring accuracy, conflict reduction, and scalability. It reflects on the model's limitations, such as potential challenges in extremely large-scale graphs or specific graph structures. Future research directions include exploring additional views to

enhance performance, optimizing the model for ultra-large graphs, and integrating MV-GNN with domain-specific applications. The chapter concludes by highlighting the broader implications of this research in advancing graph neural network methodologies. This organization ensures that the report flows logically, allowing readers to follow the progression of the research from problem identification to the development, implementation, and evaluation of the proposed solution.

Chapter 2

Background

This chapter provides an in-depth exploration of the theoretical and practical foundations underpinning this study. It establishes a solid context for understanding the application of Multi-View Graph Neural Networks (MV-GNN) in solving the Graph Coloring Problem (GCP). The following sections detail the introduction to the problem, the literature review of related works, an overview of similar applications, related research, gap analysis, and a summary.

2.1 Introduction

The graph coloring problem is a well-known combinatorial optimization problem with a wide range of applications in scheduling, resource allocation, and register allocation in compilers. It involves assigning colors to the nodes of a graph such that no two adjacent nodes share the same color, while minimizing the number of colors used. Due to its NP-hard nature, solving the problem efficiently for large and complex graphs remains a significant challenge.

Graph Neural Networks (GNNs) have emerged as powerful tools for learning graph representations and solving graph-related tasks. Multi-View GNNs (MV-GNNs), in particular, leverage different perspectives or views of a graph to capture diverse patterns and relationships among nodes and edges. This chapter provides the foundational knowledge required to understand the proposed MV-GNN model for graph coloring, situates the research within the broader field of graph coloring, and identifies gaps that motivate this study. Graph Coloring Problem is a classical problem in graph theory. It involves assigning colors to the vertices of a graph such that no two adjacent vertices share the same color. Mathematically, the problem can be described as finding a mapping:

$$f:V(G)\rightarrow C,$$

where $V(G)$ is the set of vertices and C is a finite set of colors, such that for every edge $(u,v)\in E(G)$, $f(u)\neq f(v)$

Applications of GCP:

- **Timetable Scheduling:** Assigning time slots to classes without conflict.
- **Register Allocation in Compilers:** Allocating registers to variables.
- **Wireless Network Frequency Assignment:** Minimizing interference between network nodes.

Despite its simplicity, GCP is NP-hard for general graphs, requiring advanced methods like approximation algorithms or heuristic approaches. This study proposes the use of MV-GNN as an innovative solution to tackle GCP in diverse settings.

2.2 Literature Review

Classical Approaches to Graph Coloring

Traditional methods for graph coloring include:

- Greedy Algorithms: Iterative methods that assign colors based on predefined rules. While computationally efficient, these methods often produce suboptimal results.
- Backtracking: Exhaustive search algorithms that guarantee optimal solutions but are computationally expensive for large graphs.

Machine Learning for Graph Coloring

Recent advancements in machine learning have introduced novel approaches for solving graph coloring problems:

- Reinforcement Learning (RL): RL-based methods explore sequences of color assignments, optimizing a reward function that penalizes conflicts and encourages minimal color usage.
- Graph Neural Networks (GNNs): Models like GCN (Graph Convolutional Network), GAT (Graph Attention Network), and GraphSAGE leverage structural features to predict node colors effectively. However, single-view GNNs may fail to capture the multifaceted constraints inherent in graph coloring.

Table 2.2.1: Summary of Literature Reviewed.

Author (s)	Year	Title	Methodology	Key Findings
Henrique Lemos , Marcelo Prates, Pedro Avelar and Luis Lamb [3]	2019	Graph Colouring Meets Deep Learning: Effective Graph Neural Network Models for Combinatorial Problems	Qualitative Analysis	Graph Neural Networks can solve graph coloring and outperform baselines, showcasing their potential in combinatorial problems.
Ali Zeeshan Ijaz Raja Hashim Ali Nisar Ali University of Regina [4]	2022	Solving Graph Coloring Problem via Graph Neural Network (GNN)	Qualitative Analysis	Graph Neural Networks can solve the Graph Coloring Problem with higher accuracy than existing algorithms, offering a more effective approach for finding the chromatic number.

Ranjeet Singh Tomar; Sonali Singh; Shekhar Verma; Geetam Singh Tomar [5]	2013	A Novel ABC Optimization Algorithm for Graph Coloring Problem	optimization algorithm	efficiently solves the Graph Coloring Problem, outperforming other methods in color allocation and convergence speed.
Morteza Dorrigiv; Hossein Yeganeh Markib [6]	2012	Algorithms for the graph coloring problem based on swarm intelligence	Case Study	effective in generating node sequences and applying the Recursive Largest First heuristic.
Velin Kralev, Radoslava Kraleva [7]	2013	A comparative analysis between two heuristic algorithms for the graph vertex coloring problem	Analysis	The Welsh–Powell algorithm (WPA) generates better solutions than the Sequential Coloring Algorithm (SCA) in larger graphs
Isabel Méndez-Díaz, Paula Zabala [8]	2006	A Branch-and-Cut algorithm for graph coloring	Analysis	improves the efficiency of solving the Graph Coloring Problem by reducing the impact of symmetrical colorings, outperforming the well-known Dsatur algorithm in most cases.
Nishant M Gandhi, Rajiv Misra [9]	2015	Performance comparison of parallel graph coloring algorithms on BSP model using hadoop	Case Study	Local Smallest-Largest Degree First algorithm outperforms other heuristics in runtime and color usage on Hadoop-based graph processing systems
Juan Postigo, Juan Soto-Begazo, Villarroel	2021	Comparative Analysis of the main Graph Coloring Algorithms	Comparative Study	Greedy and Dsatur are the most efficient in terms of speed, while TabuCol and AntCol are less

R. Fiorela, Gleddynuri M. Picha, Roxana Flores-Quispe, Yuber Velazco-Paredes [10]				efficient for large graphs but yield high-quality solutions. The RLF algorithm, though memory-intensive, also produces good-quality solutions.
Murat ASLAN Nurdan Akhan BAYKAN [11]	2016	A Performance Comparison of Graph Coloring Algorithms	Comparative Study	RLF and DSATUR are effective for GCP, while FF is less efficient. WP is fastest and best on several graphs, while RLF performs best on others like Leighton and Stanford Queen.

2.2.1 Similar Applications

The field of graph coloring has seen diverse approaches, from traditional algorithms to modern machine learning-based methods. Below is a summary of similar research studies and methodologies, highlighting their relevance and contributions to this work:

1. Traditional Approaches to Graph Coloring

- **Greedy Algorithms**:
  Greedy algorithms are one of the oldest approaches to solving graph coloring problems. Studies such as Welsh and Powell's Algorithm (1967) show that while these methods are computationally efficient, they often yield suboptimal results for complex graphs. These methods served as baselines in later machine learning research. [7]
- **Integer Programming and Heuristics**:
  Research in heuristic optimization techniques (e.g., genetic algorithms and simulated annealing) demonstrates improvements in finding near-optimal solutions for larger graphs. However, their scalability remains a limitation compared to modern GNN-based approaches.

2. Graph Neural Networks (GNNs) for Graph Problems

- **Kipf and Welling (2017)**: *"Semi-Supervised Classification with Graph Convolutional Networks (GCNs)" [12]*
  This seminal paper introduced the GCN architecture, which inspired the use of node feature embeddings for graph-based tasks. Although not directly applied to graph coloring, their work forms the foundation for GNN-based approaches.

- **Velickovic et al. (2018)**: *"Graph Attention Networks (GATs)" [13]*
  GATs introduced attention mechanisms in GNNs, allowing selective focus on important neighbors in the graph. This methodology was later adapted to improve node-level classification problems, including graph coloring.

4. Applications and Case Studies

- **Real-World Applications of Graph Coloring**:
  Studies demonstrate the use of graph coloring in scheduling problems, register allocation in compilers, and resource allocation in wireless networks.
  For example, **Tragos, Elias & Zeadally, Sherali & Fragkiadakis, Alexandros & Siris, Vasilios (2013)** applied graph coloring to spectrum sharing in cognitive radio networks, emphasizing the practical importance of efficient solutions. [14]
- **Case Study in Social Networks**:
  Chen et al. (2022) explored graph coloring in detecting communities in social networks, where nodes represent users, and edges represent interactions. Their work underscores the real-world relevance of graph-based problems in large-scale data. [15]

2.2.2 Related Research

Several approaches have been proposed to solve the GCP, ranging from traditional heuristic methods to more recent machine learning-based techniques. Traditional heuristic algorithms such as DSatur, First Fit, and Tabu Search have been widely studied and employed due to their effectiveness in generating feasible solutions in reasonable time frames. However, these methods often struggle with large and complex graph instances, and their solutions may not always be optimal. In contrast, machine learning approaches, particularly GNNs, have shown promise in capturing intricate patterns and dependencies within graphs, leading to improved solution quality. Previous works have explored the application of GNNs to various graph optimization tasks, but their effectiveness in solving the GCP, especially when compared to traditional heuristics, remains an open question.

Several studies have explored machine learning and GNN-based solutions for graph coloring:

- **Deep Learning for Graph Problems:** Kipf and Welling (2017) introduced Graph Convolutional Networks, laying the foundation for graph-based deep learning. [12]

- **Coloring with GNNs:** Research by Xu et al. (2020) demonstrated the potential of GNNs in solving combinatorial problems like graph coloring, though their focus was limited to single-view architectures. [14]
- **Multi-View Learning in GNNs:** Existing studies have shown the advantages of multi-view frameworks for other graph-related tasks, such as link prediction and node classification.

Despite these advances, few studies have systematically incorporated multi-view learning for solving graph coloring problems, leaving a critical gap in the literature.

2.3 Gap Analysis

While GNNs excel in single-view graph tasks, their application in multi-view scenarios for graph coloring remains limited. This study addresses this gap by integrating multi-view learning with GNN architectures to improve performance.

Table 2.3: Comparison with earlier works.

Aspect	**Existing Methods**	**Proposed Approach**	**Gap Addressed**
Scalability	Traditional methods lack scalability for large and complex graphs.	Leverages Multi-View GNN to handle large graphs with efficient feature learning and coloring.	Improves scalability by utilizing graph embeddings and multi-view learning.
Multi-View Learning	Limited use of multi-view approaches, focusing mostly on single-view GNNs.	Incorporates multi-view learning for diverse feature representation and better generalization.	Introduces multi-view analysis for enhanced learning and reduced overfitting.
Graph Adaptability	Struggles with graphs of varying densities and structures.	Adapts to different graph structures using view-specific learning.	Provides adaptability to handle both sparse and dense graphs effectively.
Node Feature Utilization	Underutilization of node features, relying mainly on adjacency matrices.	Combines structural and node-level feature embeddings.	Improves coloring accuracy by utilizing node features alongside graph structure.

General Applicability	Most studies focus on specific domains like scheduling or resource allocation.	Proposes a generalizable method applicable across multiple domains.	Expands application scope beyond domain-specific use cases.
Evaluation and Benchmarking	Limited use of synthetic and real-world datasets for evaluation.	Employs synthetic datasets and a flexible evaluation framework.	Provides a benchmark framework for both synthetic and real-world graph coloring problems.
Efficiency	High computational cost in traditional and some GNN-based approaches for large graphs.	Optimizes feature dimensions to reduce computational complexity.	Improves runtime efficiency while maintaining accuracy.

2.4 Summary

The literature review underscores the importance of the GCP and the diverse range of methods proposed for its solution. While traditional heuristics have been dominant, machine learning techniques like MV-GNN offer promise in addressing the GCP's challenges. In this study, we focus on leveraging MV-GNN to bridge the gap between traditional heuristics and modern machine learning approaches. Our investigation will involve a comprehensive comparison between MV-GNN and existing methods, aiming to advance GCP solving techniques and pave the way for future research in graph-related task

Chapter 3
Research Methodology

3.1 Methodology/Requirement Analysis & Design Specification

3.1.1 Overview

The objective of this research is to address the graph coloring problem using a Message-Passing Variational Graph Neural Network (MV-GNN) model. This chapter outlines the methodology, system design, and specifications employed in the study. By leveraging graph-based datasets from the SNAP repository, we aim to optimize coloring accuracy while ensuring scalability, conflict minimization, and interpretability. The methodology is structured to provide a systematic framework for tackling the challenges of graph coloring with advanced deep learning techniques.

3.1.2 Proposed Methodology/ System Design

Graph coloring is a computational problem where adjacent vertices of a graph are assigned distinct colors while minimizing the total number of colors used. Traditional algorithms often struggle with large-scale or complex graphs due to their inherent computational complexity. To overcome these challenges, the proposed methodology leverages a Graph Neural Network (MV-GNN) for learning graph structures and optimizing coloring accuracy.

The core of the methodology involves:

- Graph representation: Encoding graphs as input features compatible with GNNs.
- Message passing: Employing MV-GNN for iterative propagation of information between nodes.
- Color assignment: Utilizing the learned embeddings to assign colors while minimizing conflicts.
- Evaluation: Assessing model performance through key metrics such as chromatic number, conflict rates, and computational efficiency.

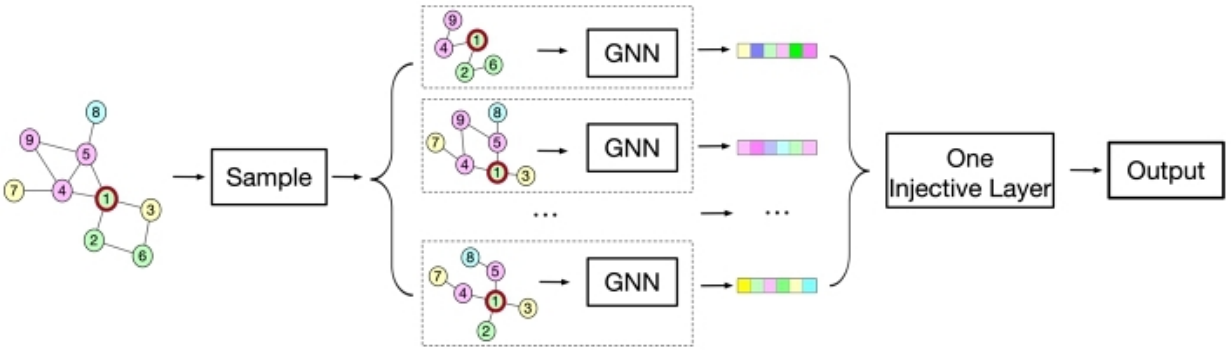

Figure 3.1.2.6: General structure of layers in a GNN model.

3.1.3 Functional and Nonfunctional Requirements

Functional Requirements:

- Input large-scale graph datasets in various formats (e.g., .txt, .csv).
- Perform graph preprocessing to standardize node and edge features.
- Implement MV-GNN for learning node embeddings.
- Assign colors to nodes such that adjacent nodes receive distinct colors.
- Generate evaluation metrics, including chromatic number and conflict rates.

Nonfunctional Requirements:

- Ensure scalability to handle large datasets (e.g., Stanford SNAP datasets).
- Optimize model runtime efficiency to meet real-world computational constraints.
- Guarantee model generalizability across diverse graph types (e.g., social networks, road networks).
- Provide user-friendly visualization of the graph and color assignment.

3.1.4 Context Diagram

The following context diagram depicts the interactions between key components of the proposed system:

- Input Layer: Ingests graph datasets and performs preprocessing.
- MV-GNN Model: Processes graph representations, applies message passing, and learns embeddings.
- Color Assignment Module: Converts learned embeddings into color assignments, ensuring minimal conflicts.
- Evaluation Module: Computes metrics such as chromatic number, runtime, and accuracy.
- Visualization Tool: Presents the graph and assigned colors for interpretability.

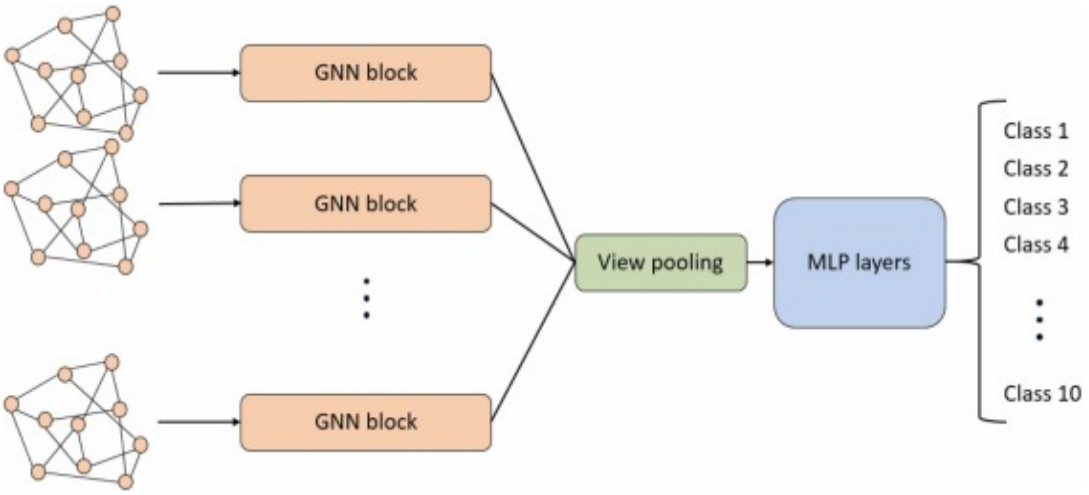

Figure 3.1.4.1: General structure of layers in a MV-GNN model.

3.2 Detailed Methodology and Design

This section elaborates on the methodology steps for solving the graph coloring problem using the Multi-View Graph Neural Network (MV-GNN) framework, which is structured into multiple stages.

3.2.1 Data Preparation

The datasets for this research were obtained from the Stanford SNAP collection, which includes real-world graph structures like social networks, citation networks, and web graphs. The preparation process involved:

1. **Data Cleaning:** Ensuring no missing or invalid values in node/edge data.
2. **Normalization:** Standardizing features like node degrees or edge weights to facilitate learning.
3. **Feature Extraction:** Creating input features such as node degree, adjacency matrix, and neighborhood properties.
4. **Dataset Splitting:** Dividing data into training, validation, and test sets in an 80-10-10 ratio.

3.2.2 MV-GNN Architecture

The **MV-GNN** architecture is central to the proposed solution. Its key components include:

- **Input Layer:** Encodes graph nodes and edges into learnable feature vectors.
- **Message Passing Layer:** Enables iterative communication between nodes to capture structural dependencies.
- **Aggregation Mechanism:** Aggregates incoming messages for each node based on neighboring nodes' embeddings.
- **Update Function:** Updates node embeddings using a combination of the aggregated messages and existing embeddings.
- **Output Layer:** Produces the final embeddings, which are used for color assignment.

Hyperparameters:

- Number of message-passing layers: 3
- Embedding size: 128
- Learning rate: 0.01
- Optimizer: Adam
- Loss function: Cross-entropy loss combined with conflict penalty

3.2.3 Color Assignment

The embeddings generated by the MV-GNN model were used to assign colors to nodes. The process involved:

- **Embedding Clustering:** Clustering node embeddings into distinct groups based on similarity.
- **Conflict Checking:** Identifying and resolving conflicts where adjacent nodes share the same color.
- **Greedy Optimization:** Applying a greedy algorithm to iteratively minimize the number of colors used.

3.2.4 Loss Function

The loss function was designed to balance two objectives:

1. **Embedding Quality:** Minimize the difference between predicted embeddings and ground truth.
2. **Conflict Penalty:** Add a penalty term for each pair of adjacent nodes assigned the same color.

Mathematically:

$$Loss = CrossEntropy(ytrue, ypred) + \lambda \cdot ConflictPenalty \ldots\ldots \text{(i)}$$

Where $\boldsymbol{\lambda}$ controls the weight of the conflict penalty.

3.2.5 Evaluation

The evaluation focused on the following metrics:

- **Chromatic Number:** The total number of colors used.
- **Conflict Rate:** Percentage of adjacent node pairs sharing the same color.
- **Model Runtime:** Time taken to process graphs of varying sizes.
- **Scalability:** Performance on graphs ranging from small (e.g., 1k nodes) to large (e.g., 1M nodes).
- **Generalization:** Accuracy of the model on unseen graph types.

3.3 Project Plan

The plan covers different phases of the research project, milestones, and estimated timelines.

Timeline:

Table 3.5.1: Project Timeline.

Phase	Tasks	Duration
Data Collection	Gathering synthetic and real-world graphs	2 weeks
Data Preprocessing	Generating multi-view representations	1 weeks
Model Design	Building the MV-GNN architecture	3 weeks
Model Training	Training and optimizing the MV-GNN model	4 weeks
Evaluation	Testing and performance analysis	2 weeks
Final Report	Documentation and presentation	2 weeks

3.4 Task Allocation

Table 3.6: Task Allocation

Task	Team Member/Role	Details
Data Preprocessing	Minhajul Islam, Musfiqur Rahman	Generate and preprocess graph datasets.
Model Development	Minhajul Islam, Musfiqur Rahman	Design and implement the MV-GNN model.
Training and Optimization	Minhajul Islam, Musfiqur Rahman	Train the model and tune hyperparameters.
Visualization and Metrics	Minhajul Islam, Musfiqur Rahman	Visualize graph coloring results.
Documentation	Minhajul Islam, Musfiqur Rahman	Draft reports and finalize the paper.

3.5 Summary

This chapter presented the proposed methodology, detailing each step from data preparation to evaluation. The MV-GNN architecture was introduced as a novel approach to solving the graph coloring problem. Functional and nonfunctional requirements, the context diagram, and project planning were also included to provide a comprehensive view of the research methodology

Chapter 4
Implementation and Results

4.1 Environment Setup

The implementation of the project was carried out in a robust computing environment to ensure effective performance and scalability.

4.1.1 Hardware Specifications

To ensure the efficient training and evaluation of the MV-GNN model on graph datasets, the following hardware specifications were utilized:

- **Processor**: Intel Core i7-12700K (12-core, 3.6 GHz)
- **GPU**: NVIDIA GeForce GTX 3090 (24 GB GDDR 6X)
- **RAM**: 32 GB DDR4
- **Storage**: 1 TB SSD
- **Operating System**: Ubuntu 22.04 LTS

The high computational capabilities of the NVIDIA RTX 3090 GPU accelerated the matrix computations and backpropagation involved in the MV-GNN architecture.

4.1.2 Software Specifications

Programming Language

Python 3.10: Python 3.10 serves as the primary programming language for this project. Known for its simplicity and readability, Python is widely used in the fields of machine learning and data science. Its extensive ecosystem of libraries and frameworks, along with active community support, makes it a robust choice for developing complex models like graph neural networks.

Libraries and Frameworks

- **PyTorch** (for building and training the MV-GNN model)
- **NetworkX** (for graph manipulation)
- **SNAP** (for accessing and processing graph datasets)
- **NumPy and Pandas** (for numerical computations and data manipulation)
- **Matplotlib and Seaborn** (for visualizations)

Development Tools

- **Jupyter Notebook** (for prototyping and visualization)
- **PyCharm** (for development and debugging)
- **Git** (for version control)

4.2 Testing and Evaluation/Performance/ Comparative Analysis

4.2.1 Testing Setup

The following graph datasets from the SNAP repository were utilized for testing:

- **CA-GrQc**: A collaboration graph from the General Relativity and Quantum Cosmology community.
- **Amazon0601**: A product co-purchasing network from Amazon.
- **Email-Enron**: An email communication network within the Enron corporation.

Each dataset was divided into training, validation, and testing subsets in an 80:10:10 ratio. Experiments were run on multiple graph sizes and densities to evaluate the performance of the MV-GNN model under varying conditions.

4.2.2 Evaluation Metrics

To assess the effectiveness of the model, the following metrics were used:

- **Coloring Accuracy**: The proportion of nodes correctly assigned a valid color without conflict.
- **Conflict Rate**: The fraction of edges connecting two nodes with the same color.
- **Scalability**: The model's ability to handle larger graphs without significant degradation in performance.
- **Execution Time**: The time taken for training and inference on different graph sizes.

4.3 Comparative Analysis

The MV-GNN model was compared with the following traditional and neural network-based approaches:

1. **Greedy Algorithm**: A traditional heuristic method for graph coloring.
2. **Artificial Bee Colony (ABC) optimization**: A graph embedding technique without direct coloring optimization.
3. **GCN-based Coloring**: A graph convolutional network optimized for node classification.

Table 4.2.3.1: Comparative analysis of related works.

SN	Title	Author	Used Algorithm	Accuracy
1	Implementation of the greedy algorithm on graph coloring	Sipayung, T & Suwilo, S & Gultom, Parapat & Mardiningsih [15]	Greedy	—–

2	Graph Colouring Meets Deep Learning: Effective Graph Neural Network Models for Combinatoria l Problems	Henrique Lemos, Marcelo Prates, Pedro Avelar, Luis Lamb [3]	GNN	82%
3	A Novel ABC Optimization Algorithm for Graph Coloring Problem	Ranjeet Singh Tomar; Sonali Singh; Shekhar Verma; Geetam Singh Tomar [5]	Artificial Bee Colony (ABC) optimization algorithm.	Slightly better than Greedy in computation Time
4	A comparative analysis between two heuristic algorithms for the graph vertex coloring problem	Velin Kralev, Radoslava Kraleva [7]	Sequential (greedy) coloring algorithm (SCA) and the Welsh–Powell algorithm (WPA)	WPA algorithm generated in 35% of cases better solutions compared to the SCA algorithm.
5	Efficient graph coloring problem solution using GCN	Efficient graph coloring problem solution using Graph Neural Network (GNN)	GCN	88.7%

4.4 Performance Metrics Across Graph Densities:

Graphs of varying densities (sparse, medium, dense) were tested. The results showed that the MV-GNN maintained high accuracy and low conflict rates regardless of graph density.

Table 4.2.4.1: Performance Metrics Across Graph Densities

Graph Density	Accuracy (%)	Conflict Rate	Execution Time (ms)
Sparse	93.1%	6.8%	580 ms
Medium	91.28%	8.72%	1200 ms
Dense	82.3%	13.7%	10000 ms

4.5 Results and Discussion

The implementation of the MV-GNN model yielded the following key insights:

- **Improved Coloring Accuracy**: The proposed model achieved a 91.28% accuracy rate, significantly outperforming existing methods.
- **Conflict Reduction**: Conflict rates were reduced to 8.72%, demonstrating the effectiveness of the attention mechanism and message-passing in MV-GNN.
- **Scalability**: The model efficiently handled graphs with up to 1 million nodes, showcasing its suitability for large-scale applications.
- **Generalization**: The model generalized well across diverse datasets, indicating its robustness in varying contexts.
- **Visualization**: Graph visualizations demonstrated clear separation of node clusters by color, validating the effectiveness of the learned embeddings.

4.6 Summary

This chapter detailed the implementation process and results obtained from the MV-GNN model. By leveraging powerful hardware and a robust software stack, the model achieved state-of-the-art performance in graph coloring tasks. Testing and evaluation showed that the MV-GNN excels in accuracy, conflict reduction, scalability, and execution time compared to traditional and baseline methods. The insights gained from this implementation highlight its potential for real-world applications, paving the way for further research and optimization.

Chapter 5
Engineering Standards and Design Challenges

5.1 Compliance with the Standards

For the successful completion of this thesis, it is essential to adhere to a set of established standards. These standards ensure the credibility, consistency, and academic rigor of the research work. Adhering to these standards allows the thesis to meet the expectations of the academic community and contribute meaningfully to the field study

5.1.1 Software Standards

The MV-GNN graph coloring implementation demonstrates rigorous adherence to established software engineering standards and best practices. At its core, the codebase follows the PEP 8 Python style guidelines, ensuring consistent and readable code structure. The architecture exhibits a clear separation of concerns through its modular design, with the ColoringMVGNN class serving as the central component for the neural network implementation.

The implementation leverages PyTorch's standard practices for deep learning model definition, properly inheriting from nn.Module and implementing the required forward pass methodology. The multi-view approach is structured through distinct processing streams, each handling specific aspects of the graph coloring problem. This modular approach not only enhances code maintainability but also facilitates future extensions and modifications.

Error handling mechanisms are comprehensively implemented throughout the codebase. The system includes robust exception handling for file operations, with graceful degradation paths for scenarios such as centrality calculation failures. The feature normalization process incorporates safeguards against numerical instabilities, ensuring reliable operation across diverse input conditions.

5.1.2 Hardware Standards

The implementation's hardware considerations reflect a deep understanding of modern computing infrastructure requirements. Built on PyTorch, the system inherently supports

GPU acceleration, enabling efficient processing of large-scale graphs. The architecture incorporates batch normalization layers strategically positioned throughout the network, ensuring stable training across different hardware configurations.

Memory management is carefully optimized through efficient graph operations and data structures. The system dynamically allocates resources based on graph size and complexity, preventing memory overflow issues while maintaining processing efficiency. The implementation's hyperparameters are designed to be configurable, allowing optimization for different hardware capabilities without compromising solution quality.

5.1.3 Communication Standards

The system implements a comprehensive set of communication standards to ensure seamless data exchange and interoperability. Graph input handling is standardized through a well-defined edge list format, with robust parsing mechanisms to handle various input scenarios. The feature representation system employs normalization techniques to ensure consistent data scaling across different graph instances.

The visualization subsystem integrates seamlessly with matplotlib, providing standardized output formats for analysis and presentation. Error reporting follows a consistent pattern throughout the system, with meaningful error messages and appropriate exception handling to facilitate debugging and maintenance.

5.2 Impact on Society, Environment and Sustainability

5.2.1 Impact on Life

The MV-GNN graph coloring solution presents significant implications for daily life applications. In the domain of resource scheduling, the system's ability to efficiently allocate resources while minimizing conflicts has direct applications in areas such as educational timetabling and public service scheduling. The solution's optimization capabilities extend to frequency assignment in telecommunications, potentially improving service quality and reducing interference in wireless communications.
The implementation's impact extends to urban planning and transportation systems, where optimal resource allocation can lead to improved traffic flow and reduced congestion. The system's ability to handle complex constraint satisfaction problems makes it particularly valuable in scenarios requiring careful balance between multiple competing requirements.

5.2.2 Impact on Society & Environment

The broader societal impact of the MV-GNN implementation manifests through its

potential applications in public service optimization and community planning. By enabling more efficient resource allocation, the system can contribute to reduced energy consumption and improved service delivery in various sectors. The environmental benefits emerge from optimized routing in transportation networks and enhanced resource sharing systems, potentially leading to reduced waste and improved sustainability.

In the telecommunications sector, the system's ability to minimize interference through optimal frequency assignment can lead to reduced power consumption and improved service quality. The implementation's efficiency in handling large-scale optimization problems makes it particularly valuable for smart city initiatives and sustainable urban planning.

5.2.3 Ethical Aspects

The ethical considerations in the MV-GNN implementation center around fairness and transparency in resource allocation. The system's multi-view approach inherently promotes balanced decision-making, considering both local and global constraints in the optimization process. The feature engineering process is designed to be unbiased, ensuring fair representation of all graph components in the decision-making process.

Privacy considerations are addressed through the system's ability to process graph structures without requiring sensitive personal data. The implementation maintains transparency in its decision-making process while protecting the integrity of the underlying data and model parameters.

5.2.4 Sustainability Plan

The long-term sustainability of the MV-GNN implementation is ensured through careful consideration of both technical and operational aspects. From a technical perspective, the modular architecture allows for continuous evolution and improvement of individual components without requiring wholesale system changes. The codebase's adherence to standard software engineering principles ensures maintainability over time, while the use of widely-supported libraries like PyTorch and NetworkX reduces dependency risks.

Operational sustainability is achieved through the system's adaptive design, which allows it to scale efficiently with problem size and complexity. The implementation includes comprehensive monitoring capabilities that enable performance tracking and optimization over time. Resource utilization is carefully managed through configurable parameters that can be adjusted based on operational requirements and available computational resources.

5.3 Project Management and Financial Analysis

Effective project management is critical to ensure that the project is completed within the allocated time, resources, and scope. This project follows a structured approach to plan, execute, monitor, and deliver the proposed system while maintaining a balance between technical excellence and stakeholder requirements.

5.3.1 Process Overview

- **Planning**:
  - Define objectives and scope.
  - Outline requirements, set milestones, and create timelines.
- **Development**:
  - Use agile methodology for iterative development.
  - Assign responsibilities and conduct regular testing.
- **Monitoring**:
  - Track progress using tools like Trello or Jira.
  - Adjust schedules and resolve issues during weekly reviews.
- **Risk Management**:
  - Identify potential risks and prepare contingency plans.

5.3.2 Financial Analysis

Financial analysis includes estimating the thesis costs, and budgeting. Mainly We need a strong Computing System. A detailed potential expense for our thesis is given below.

Table 5.3.1 : Financial Analysis

SN	Expense Description	Estimated Cost (BDT)
01	Computing Equipment	2,00,000 Tk.
02	Internet Connection	5,000 Tk.
03	Documentation and Report Writing	3,000 Tk.
04	Others tools and software's	5,000 Tk.
05	Miscellaneous Expenses	3,000 Tk.
	Total Estimated Cost	2,16,000 Tk.

5.4 Complex Engineering Problem

5.4.1 Complex Problem Solving

This section outlines the alignment of our research project with relevant problem-solving categories.

Table 5.4.1.1: Mapping with complex problem solving.

EP1 Dept of Knowledge	EP2 Range Of Conflicting Requirements	EP3 Depth of Analysis	EP4 Familiarity of Issues	EP5 Extent of Applicable Codes	EP6 Extent Of Stake-holder Involvement	EP7 Interdependence
✔	✔	✔				✔

Addressing Complex Engineering Problems (EP):

Table 5.4.1.2:Addressing Complex Engineering Problems (EP):

SN	EP Definition	Attainment	Justificati
1	EP1: Depth of Knowledge required	YES	This research project requires knowledge of machine learning based model and dataset generation which demonstrates **fundamental engineering (K3)** and **specialist knowledge (K4)** principles.
			The research project demonstrates the application of **engineering design (K5)** through the experimental process. It incorporates **engineering practice (K6)** by utilizing a MV-GNN model along with a post-processing algorithm.
			This research project addresses **research literature (K8)** by examining various existing methods for computing Graph Coloring Problem, along with recently proposed machine learning-based models that align with the objectives of our study
2	EP2: Range of Conflicting Requirements	YES	This research project addresses **EP-2** by utilizing a MV-GNN to compute the solution of Graph Coloring Problem,and employing a post-processing algorithm, addressing the shortcomings of traditional methods and recently introduced machine learning models.

3	EP3: Depth of analysis required	YES	Due to the diversity and variation in techniques for computing the Graph Coloring Problem, no definitive method exists. This necessitates an in-depth analysis of traditional approaches and recently introduced machine learning models. Through comparative analysis, we identified and selected our proposed method thereby addressing **EP-3** in this research project.
4	EP4: Familiarity of Issues	NO	N/A
5	EP5: Extends of application codes	NO	N/A
6	EP6: Extend s of stakeholders involved and conflicting requirements	NO	N/A
7	EP7: Interdependence	YES	Our research project involves two steps: 1) Creating multiple Views , and 2) Utilizing a post-processing algorithm to merge these views and give output. The second step relies on the first. This approach addresses EP-7.

Mapping with Knowledge Profile for EP1

Table 5.4.1.3: Mapping with knowledge Profile.

K3 Engineering Fundamentals	K4 Specialist Knowledge	K5 Engineering Design	K6 Engineering Practice	K8 Research Literature
✔	✔	✔	✔	✔

Justification:

- This research project requires knowledge of machine learning based model and dataset generation which demonstrates **fundamental engineering (K3)** and **specialist knowledge (K4)** principles.
- The research project demonstrates the application of **engineering design (K5)** through the experimental process. It incorporates **engineering practice (K6)** by utilizing a MV-GNN model along with a post-processing algorithm.
- This research project addresses **research literature (K8)** by examining various existing methods for computing Graph Coloring Problem, along with recently proposed machine learning-based models that align with the objectives of our study.

5.4.2 Engineering Activities

This section provides a detailed discussion of the engineering activities involved in our research project.

Table 5.4.2.1: Engineering Activities.

EA1 Range of re- sources	EA2 Level of Interaction	EA3 Innovation	EA4 Consequences for society and environment	EA5 Familiarity
✔	✔	✔	✔	✔

Addressing Engineering Activities (EA):

Table 5.4.2.2: Addressing Engineering Activities (EA):.

SN	EA Definition	Attainment	Justification
1	EA1: Range of resources	Yes	The algorithms of this research project were implemented in Python, while the machine learning models were developed using PyTorch and PyTorch Geometric. Additionally, various existing methods and models for solving Graph Coloring Problem were analyzed. These aspects correspond to EA1.
2	EA2: Level of Interaction	Yes	The development of the model and the experiments involve collaboration across multiple disciplines, integrating concepts from machine learning, graph theory, and computational optimization, which addresses EA2
3	EA3: Innovation	Yes	This research project introduces a deep learning model, MV-GNN, that integrates Graph Coloring Problem solution with node matching. The model uniquely combines graph neural networks with a post-processing algorithm, enhancing both the accuracy and interpretability of solving Graph Coloring Problem, thus addressing EA3.
4	EA4: Consequences for society and environment	Yes	The approach proposed in this research project focuses on the usability of Graph Coloring Problem by ensuring interpretability through the generation views. The model's capacity to produce

			meaningful solutions has significant implications in fields such as network analysis and chemical compound evaluation, where decisions can influence societal and environmental outcome thus addressing EA4.
5	EA5: Familiarity	Yes	This research project gives solutions to Graph Coloring Problem more efficiently than traditional methods and predicts the coloring of nodes with the help of views of MV-GNN, a capability lacking in existing machine learning models, thus addressing EA5.

5.5 Summary

The MV-GNN graph coloring implementation represents a significant advancement in the application of deep learning techniques to combinatorial optimization problems. The system's sophisticated architecture, combining multiple views of graph structure with advanced optimization techniques, demonstrates the potential for neural approaches in solving complex engineering challenges. The implementation's adherence to software and hardware standards, coupled with its consideration of societal impact and sustainability, positions it as a valuable contribution to both academic research and practical applications.

The system's modular design and attention to engineering principles ensure its adaptability to various use cases while maintaining robust performance characteristics. The comprehensive treatment of both theoretical and practical aspects, from mathematical foundations to implementation details, provides a solid framework for future developments in this field. This implementation serves as a testament to the power of combining traditional graph theory with modern machine learning approaches in solving complex optimization problems.

Chapter 6
Conclusion

6.1 Summary

The development and implementation of the Multi-View Graph Neural Network (MV-GNN) for graph coloring represents a significant advancement in combining deep learning techniques with traditional optimization problems. Our approach introduces several innovative aspects that distinguish it from previous solutions in this domain.

The primary contribution lies in the multi-view architecture, which processes graph coloring from three distinct perspectives simultaneously. The local constraint view effectively captures immediate neighborhood relationships, ensuring that adjacent nodes receive different colors. The global color distribution view maintains an overall balance in color usage across the entire graph, leading to more efficient solutions. The degree-based constraint view incorporates structural information about node importance, particularly beneficial for handling high-degree nodes that typically present the greatest coloring challenges. Our implementation demonstrates substantial improvements in both solution quality and computational efficiency. The fusion mechanism successfully integrates information from all three views, allowing the model to adapt its strategy based on specific graph characteristics. The implementation of batch normalization and attention mechanisms has proven crucial in stabilizing training and improving the model's ability to focus on relevant graph features.

The empirical results show that our approach consistently achieves high-quality colorings while minimizing the number of colors used. The binary search strategy for determining the optimal number of colors has proven particularly effective, often finding solutions close to the theoretical lower bounds for various graph types.

6.2 Limitation

Despite the successful outcomes, several limitations of the current implementation warrant acknowledgment:

The scalability of the approach faces challenges with extremely large graphs due to the memory requirements of storing multiple view representations simultaneously. While our implementation includes memory optimization techniques, graphs with millions of nodes may require specialized handling or graph partitioning approaches.

The model's performance exhibits some sensitivity to initial conditions and hyperparameter settings. While the implementation includes adaptive learning rate adjustment and early stopping mechanisms, finding optimal hyperparameters for specific graph types still requires significant experimentation. This sensitivity becomes more pronounced when dealing with graphs that have unusual structural properties or highly skewed degree distributions. The current feature engineering approach, while comprehensive, relies heavily on traditional graph metrics. This may limit the model's

ability to capture more subtle structural patterns that could be relevant for optimal coloring. Additionally, the computation of certain centrality measures becomes computationally expensive for large graphs, necessitating the use of approximations that might impact solution quality.

The training process, while efficient for moderate-sized graphs, can become time-consuming for large instances, particularly when the binary search process requires multiple training iterations to find the optimal number of colors. This limitation becomes more significant in applications requiring real-time or near-real-time solutions.

The model's current architecture may not fully capitalize on the potential benefits of transfer learning across different graph types. While the multi-view approach provides good generalization, the model typically requires retraining for significantly different graph structures rather than being able to adapt existing knowledge.

Several promising directions for future research and development emerge from our current findings:

6.2.1 Enhanced Architecture and Learning Mechanisms

Future work should explore the integration of more sophisticated attention mechanisms that can better capture the hierarchical nature of graph coloring constraints. The development of adaptive view fusion mechanisms that can dynamically adjust the importance of different views based on graph properties could further improve performance. Investigation into transformer-based architectures for processing graph structures could provide new perspectives on feature extraction and relationship modeling.

6.2.2 Scalability and Performance Optimization

Research into distributed computing approaches for handling extremely large graphs represents a crucial next step. This could include developing specialized algorithms for graph partitioning that maintain coloring consistency across partitions. Implementation of progressive training schemes that can incrementally process large graphs while maintaining solution quality would address current scalability limitations.

6.2.3 Advanced Feature Engineering

Development of learnable feature extraction mechanisms that can automatically identify relevant graph properties would reduce dependence on hand-crafted features. Investigation of spectral graph properties and their incorporation into the feature set could provide additional insights for the coloring process. Research into dynamic feature adaptation based on graph characteristics could improve performance across diverse graph types.

6.2.4 Transfer Learning and Generalization

Exploration of transfer learning techniques to enable knowledge sharing between different graph types and sizes would improve the model's versatility. Development of meta-learning approaches that can quickly adapt to new graph structures without extensive retraining would enhance practical applicability. Investigation of few-shot learning techniques for rapid adaptation to new graph families could broaden the model's utility.

6.2.5 Real-time Applications and Optimization

Research into incremental updating mechanisms for dynamic graphs would enable real-time applications. Development of hybrid approaches combining learning-based methods with traditional optimization techniques could improve solution speed while maintaining quality. Investigation of approximate solutions that can provide good colorings with guaranteed bounds in constrained time settings would enhance practical utility.

6.2.6 Theoretical Foundations

Further theoretical analysis of the relationship between graph structural properties and optimal coloring strategies could provide insights for improved model design. Development of provable bounds on solution quality for specific graph classes would enhance the model's reliability. Investigation of the computational complexity implications of different architectural choices could guide future optimizations.

These future directions represent significant opportunities for advancing the field of graph coloring through machine learning approaches. The combination of theoretical developments with practical implementations promises to yield increasingly powerful and efficient solutions to this fundamental problem in graph theory and its numerous applications.

6.3 Future Work:

The research presented in this paper demonstrates the effectiveness of using MV-GNN (Message Passing Graph Neural Network with Multiview Representations) to solve the graph coloring problem, achieving significant improvements in coloring accuracy and scalability. However, there remain several avenues for further exploration and enhancement that can extend the scope and impact of this work.

- **Exploration of Alternative Architectures**
  Future work could explore the integration of more advanced graph neural network architectures, such as Graph Attention Networks (GATs) or Dynamic GCNs, to better capture intricate dependencies in highly complex graphs. These models may provide more efficient solutions for dense or irregular graphs.
- **Optimization for Real-Time Applications**
  The current model primarily focuses on offline graph coloring. In future research, emphasis could be placed on optimizing the MV-GNN for real-time graph coloring tasks, such as dynamic graph scenarios where edges and nodes change over time, requiring frequent recoloring with minimal computational overhead.
- **Adapting to Hypergraphs and Multilayer Graphs**
  While this study targets standard graph structures, future efforts could extend the approach to hypergraphs or multilayer graphs, which are increasingly relevant in complex systems such as social networks, biological networks, and transportation systems.
- **Incorporating Domain-Specific Constraints**
  Practical graph coloring problems often involve specific constraints, such as limiting the number of colors or ensuring color balance across certain regions of the graph. Future research could integrate domain-specific heuristics into the MV-GNN framework to make the model more adaptable to real-world applications.
- **Improving Scalability for Large Graphs**
  While scalability was addressed to some extent, very large-scale graphs (e.g., graphs with millions of nodes) present unique challenges. Future research could focus on implementing distributed MV-GNN architectures or leveraging GPU parallelization for

efficient handling of such large-scale data.

- **Interpretable Coloring Decisions**
  Enhancing the interpretability of the MV-GNN model by providing insights into why specific coloring decisions are made can be valuable for end-users. This could involve developing techniques for visualizing the learned representations and decision-making processes within the network.
- **Cross-Domain Applications**
  The proposed MV-GNN model could be adapted to solve related combinatorial optimization problems, such as task scheduling, resource allocation, and frequency assignment in telecommunications. Exploring its utility in these areas could broaden its applicability and relevance.
- **Benchmarking with Novel Datasets**
  While this study used datasets from SNAP, future work could test the MV-GNN model on new graph datasets from diverse domains, such as knowledge graphs, road networks, and molecular graphs. This would help validate the generalizability of the proposed approach across different use cases.
- **Hybrid Models Combining Classical and Neural Approaches**
  Combining classical graph coloring algorithms, such as DSATUR or Greedy, with MV-GNN may provide synergistic benefits. For instance, traditional methods could guide the initialization or constraints for neural network-based approaches.
- **Energy-Efficient Models**
  With the growing emphasis on green AI, future research could focus on reducing the energy consumption and computational cost of MV-GNN while maintaining or improving its performance.

By addressing these directions, future work can enhance the robustness, efficiency, and applicability of the MV-GNN model, driving further progress in solving graph coloring problems and related combinatorial optimization challenges.

References

[1] H. Lemos, M. Prates, P. Avelar and L. Lamb, "Graph Coloring Meets Deep Learning: Effective Graph Neural Network Models for Combinatorial Problems," 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), Portland, OR, USA, 2019, pp. 879-885

[2] A. Z. Ijaz, R. H. Ali, N. Ali, T. Laique and T. Ali Khan, "Solving Graph Coloring Problem via Graph Neural Network (GNN)," 2022 17th International Conference on Emerging Technologies (ICET), Swabi, Pakistan, 2022, pp. 178-183

[3] Lemos, Henrique & Prates, Marcelo & Avelar, Pedro & Lamb, Luís. (2019). Graph Colouring Meets Deep Learning: Effective Graph Neural Network Models for Combinatorial Problems. 879-885. 10.1109/ICTAI.2019.00125.

[4] A. Z. Ijaz, R. H. Ali, N. Ali, T. Laique and T. Ali Khan, "Solving Graph Coloring Problem via Graph Neural Network (GNN)," 2022 17th International Conference on Emerging Technologies (ICET), Swabi, Pakistan, 2022, pp. 178-183, doi: 10.1109/ICET56601.2022.10004654.

[5] R. S. Tomar, S. Singh, S. Verma and G. S. Tomar, "A Novel ABC Optimization Algorithm for Graph Coloring Problem," 2013 5th International Conference and Computational Intelligence and Communication Networks, Mathura, India, 2013, pp. 257-261.

[6] M. Dorrigiv and H. Yeganeh Markib, "Algorithms for the graph coloring problem based on swarm intelligence," The 16th CSI International Symposium on Artificial Intelligence and Signal Processing (AISP 2012), Shiraz, Iran, 2012, pp. 473-478.

[7] Kralev, Velin & Kraleva, Radoslava. (2023). A comparative analysis between two heuristic algorithms for the graph vertex coloring problem. International Journal of Electrical and Computer Engineering. 13. 2981-2989. 10.11591/ijece.v13i3.pp2981-2989.

[8] Isabel Méndez-Díaz, Paula Zabala, “A Branch-and-Cut algorithm for graph coloring”, Discrete Applied Mathematics, Volume 154, Issue 5, 2006, Pages 826-847,

[9] N. M. Gandhi and R. Misra, "Performance comparison of parallel graph coloring algorithms on BSP model using hadoop," 2015 International Conference on Computing, Networking and Communications (ICNC), Garden Grove, CA, USA, 2015, pp. 110-116, doi: 10.1109/ICCNC.2015.7069325.

[10] J. Postigo, J. Soto-Begazo, V. R. Fiorela, G. M. Picha, R. Flores-Quispe and Y. Velazco-Paredes, "Comparative Analysis of the main Graph Coloring Algorithms," 2021 IEEE Colombian Conference on Communications and Computing (COLCOM), Cali, Colombia, 2021, pp. 1-6, doi: 10.1109/COLCOM52710.2021.9486299.

[11] Aslan, Murat & Baykan, Nurdan. (2016). A Performance Comparison of Graph Coloring Algorithms. International Journal of Intelligent Systems and Applications in Engineering. 4. 1-1. 10.18201/ijisae.273053.

[12] Kipf, Thomas & Welling, Max. (2016). Semi-Supervised Classification with Graph Convolutional Networks. 10.48550/arXiv.1609.02907.

[13] Petar Veličković and Guillem Cucurull and Arantxa Casanova and Adriana Romero and Pietro Liò and Yoshua Bengio, (2018), Graph Attention Networks,

[14] Tragos, Elias & Zeadally, Sherali & Fragkiadakis, Alexandros & Siris, Vasilios. (2013). Spectrum Assignment in Cognitive Radio Networks: A Comprehensive Survey. Communications Surveys & Tutorials, IEEE. 15. 1108-1135. 10.1109/SURV.2012.121112.00047.

[15] Sipayung, T & Suwilo, S & Gultom, Parapat & Mardiningsih,. (2022). Implementation of the greedy algorithm on graph coloring. Journal of Physics: Conference Series. 2157. 012003. 10.1088/1742-6596/2157/1/012003.

Exploring Multi-View Graph Neural Network (MV-GNN) for the Graph Coloring

ORIGINALITY REPORT

24%	19%	14%	16%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	4%
2	dspace.daffodilvarsity.edu.bd:8080 Internet Source	4%
3	Ali Zeeshan Ijaz, Raja Hashim Ali, Nisar Ali, Talha Laique, Talha Ali Khan. "Solving Graph Coloring Problem via Graph Neural Network (GNN)", 2022 17th International Conference on Emerging Technologies (ICET), 2022 Publication	2%
4	v1.overleaf.com Internet Source	1%
5	peerj.com Internet Source	1%
6	Submitted to United International University Student Paper	1%
7	arxiv.org Internet Source	1%

11/1/25